

AD-A036 524

CLEMSON UNIV S C COLL OF ENGINEERING
A SYSTEMS ANALYSIS OF WATER QUALITY SURVEY DESIGN. APPENDIX III--ETC(U)
AUG 75 L C WILCOX, B E GILLILAND

F/G 13/2

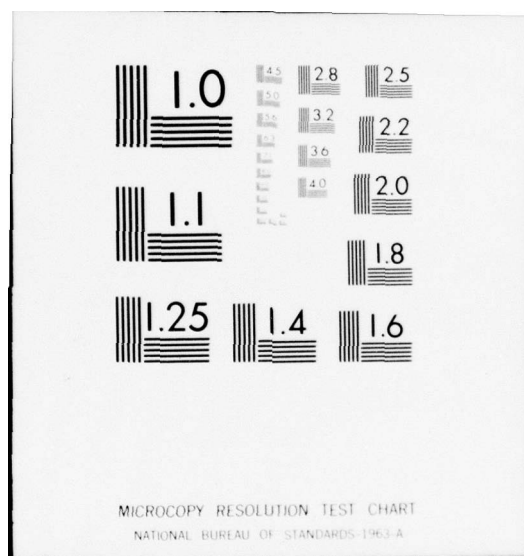
DADA17-72-C-2152

NL

UNCLASSIFIED

1 of 2
ADA036524





ADA036524

⑨ Final Rept.

12
B.S.

⑥

A SYSTEMS ANALYSIS OF WATER QUALITY SURVEY DESIGN.

FINAL REPORT
APPENDIX III
DOCUMENTATION
SURVEY PLANNING PROGRAM PROGRAMMER'S MANUAL.

⑪ AUG 1975

⑫ 147p.

⑩
Lyle & C. Wilcox,
Bobby & E. Gilliland,

SUPPORTED BY

Ralph & W. Gilchrist
Thomas & M. Keivath

U.S. ARMY MEDICAL RESEARCH AND DEVELOPMENT COMMAND
WASHINGTON, D.C. 20314

⑮
CONTRACT NO. DA DA 17-72-C-2152

PUBLICATION BY
ENGINEERING RESEARCH
CLEMSON UNIVERSITY
CLEMSON, S.C.

D D C
RECEIVED
MAR 8 1977
D

408039

DISTRIBUTION STATEMENT A

Approved for public release;
Distribution Unlimited

ACCESSION for	
REF	White Section <input checked="" type="checkbox"/>
DDC	Buff Section <input type="checkbox"/>
PREPARED BY DATE	
PROJECT CODES SPECIAL	

A

A SYSTEMS ANALYSIS OF WATER QUALITY SURVEY DESIGN

Documentation of the Survey Planning
Computer Program. A Programmer's Manual

AUTHORS: Dean L.C. Wilcox, Dr. B.E. Gilliland, Dr. R.W. Gilchrist, Dr. T.M. Keinath

August 31, 1975

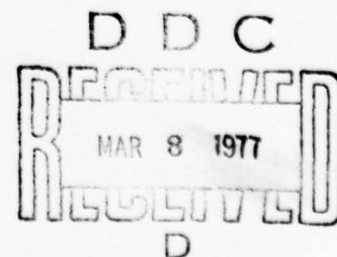
Supported By

U.S. Army Medical Research and Development Command
Washington, D.C. 20314

Contract No. DA-DA-17-72-C-2152
Clemson University
Clemson, S.C. 29631

This document has been approved for public
release; distribution unlimited.

The finding of this report are not to be construed as an official Department
of the Army position unless so designated by other authorized documents.



survey, conducting the survey and reporting on the survey were included in the study.

The site modeling program models the manufacturing processes which contribute pollutants to the system, models the sewer system, and models the treatment system including acid or caustic neutralization, settling ponds, and domestic treatment. The inputs to the model are the production levels of the manufacturing processes and the outputs are the predicted pollutant measurement values at each possible measure point in the system.

The resource matching program accepts data defining proposed measurements and matches these against the available time, manpower, and equipment. The output lists the pollutant to be measured at each measure point, the total commitment of time for each analyst and for each piece of equipment. Note is made of any overcommitment of manpower or equipment.

The model refinement or updating program accepts measurements taken during a preliminary survey or during a regular survey and computes suggested new parameters for the process models.

The indicator model program evaluates the performance of sanitary treatment facilities.

The program uses design data, data from the operating log and/or data generated during the survey and computes key operational characteristics. Comparing these with desirable values as cited in design books and manuals will give the survey planner insight into the operation of the system and suggest the need for more survey measurements or the need for changes in operation.

A system was developed for automatic instrumentation of pH, conductivity, and other parameters which use strip chart recordings. Interface hardware was selected and purchased and interface software was developed for direct connection to a digital computer.

A data handling system was developed for use during and after the survey. A PDP8-OS/8 and peripheral equipment was purchased. Software was developed to perform data handling functions and to direct the user in application of the software. The program accepts raw data from the analytical chemist and performs data conversions, transcriptions, and data logging functions. Output is available in several forms as may be needed for various reports during and at the end of the survey.

Recommendations are: the survey planner should obtain sufficient data in a preliminary survey to model and analyze the site; measurements should be automated to the maximum extent possible; data handling should be delegated to the computer when the operations are well defined and repetitive. The programs, software and hardware included here will assist the survey planner in following these recommendations and design a more effective survey.

Unlimited

TABLE OF CONTENTS

Title Page	i
Table of Contents	ii
Introduction	1
Changing Program Dimensions	2
MAIN	5
Subroutine	
PROCES	7
PROCESS LIBRARY	10
START	11
CHK 1	13
TOP	16
NEUTRA	22
SETTLE	27
DOME	29
FLAG	33
CHK 2	36
LEVEL	38
RM	45
CONCK	51
TO	56
EQCHEC	57
VSCHEC	59
CNCHEC	61
ADD	63
SET	65
PIEQCH	67

SUBT	70
INFORM	71
SAMPLE	74
PRPAR	78
USECT	82
TICHAN	83
CORRCT	84

FLOW CHARTS

Composite Program Flow Chart	91
PROCESS MODEL	92
TOP	96
NEUTRA	105
SETTLE	113
DOVE	116
RM	118
CONCK	122
PIEQCH	123
CORRCT	126

INTRODUCTION

This manual was written to assist a programmer in modifying the Survey Planning Program. The documentation given here is sufficiently complete so that an experienced programmer should be able to add to, delete or alter parts of this program.

For each subroutine the variable names and variable notations are defined. The function of each statement or small group of statements is explained. Flow diagrams are included to depict the logic relationship among statements and/or subroutines.

CHANGING SYSTEM PLANNING PROGRAM DIMENSIONS

The program and subroutines have been written so that only cards in the MAIN program need to be changed to redimension the system. The quantities that can be varied are: the number of pollutant parameters, the number of sources (processes), the number of branches, the number of pieces of equipment.

The specific items in the MAIN program which must be changed to redimension the system are given in the following paragraphs:

To handle more than 25 sources (where i = number of sources) in the MAIN program,

change MSORS = 25 to MSORS = i and in

the DIMENSION statement the arrays to modify are:

XNAME (25)		XNAME (i)
YNAME (25)		YNAME (i)
FLOW (25)		FLOW (i)
SPLIST (25,25)	CHANGE TO	SPLIST (i ,25)
CAP (25)		CAP (i)
A (25,25)		A (i ,25)
C (25,25)		C (i ,25)
P (25,25)		P (i ,25)
SUM (25)		SUM (i)
SUMA (25)		SUMA (i)

To handle more than 25 branches (where n = number of branches) in the program,

change MBRNC = 25 to MBRNC = N

and in the DIMENSION Statement the arrays to modify are:

A (25,25)		A (25, n)
Y (25,25)		Y (n , 25)
X (25,25)		X (n ,25)
EFF (25)		EFF (n)
NBRNCH (25,2)	CHANGE TO	NBRNCH (n ,2)
IBN (51,26)		IBN (51, $n+1$)
PC (25,25,25)		PC (25,25, n)
AMAR (25,27)		AMAR (25, $n+2$)

NALOW (25,25)
 BRN (25)
 BRANCH (25,25)
 NROUT (25)
 NFLOW (25)

NALOW (25,n)
 BRN (n)
 BRANCH (n,25)
 NROUT (n)
 NFLOW (n)

To handle more than 25 parameters (m = number of parameters),
 in the MAIN program,

change MPARM = 25 to MPARM = m

and in the DIMENSION statement the arrays to modify are:

SPLIST (25,25)		SPLIST (25,m)
C (25,25)		C (25,m)
Y (25,25)		Y (25,m)
X (25,25)		X (25,m)
P (25,25)		P (25,m)
NPLIST (25)		NPLIST (m)
POLN (25,5)		POLN (m,5)
NTEMP (25)		NTEMP (m)
SAMFRE (25)		SAMFRE (m)
NMA (25)		NMA (m)
MENAME (25,3,5)	CHANGE TO	MENAME (m,3,5)
PCRM (25,3)		PCRM (m,3)
PMDATA (25,3,4,5)		PMDATA (m,3,4,5)
IDO (25,25)		IDO (m,25)
PM (25,25,3)		PM (m,25,3)
AMAR (25,27)		AMAR (m,27)
PC (25,25,25)		PC (m,25,25)
NALOW (25,25)		NALOW (m,25)
EQUSED (25,75)		EQUSED (m,75)
NSET (25,3)		NSET (m,3)
FLGPT (25)		FLGPT (m)
RANK (25)		RANK (m)
BRANCH (25,25)		BRANCH (m)
ELE (25)		ELE (m)

To handle more than 75 pieces of equipment (j = number of equipment)
 in the MAIN program,

change MEQ = 75 to MEQ = j

and in the DIMENSION statement:

change VNSP (75) to VNSP (j)

SMEQTI (75)		SMEQTI (j)
EQUSED (25,75)	CHANGE TO	EQUSED (25, j)

EU (75)
EQTIME (75)
EQNAME (75,5)

EU (j)
EQTIME (j)
EQNAME (j,5)

It should be noted that these are the only changes required on the array dimensions. There is no need to make modifications in the sub-routines. Also, it is very important that the array parameters MSORS, MBRNC, MPARM, MEQ, MBRP1, and MP2 and the array subscripts correspond exactly. For example, if MPARM = 31, then every array that is affected by an increase in parameter capability must have "31" appearing in the proper subscript position as explained in that section. Care should also be exercised that every array has its proper subscript changed. Any omission may lead to errors that may not be immediately apparent.

MAIN - This program calls the subroutines LEVEL, START, CORRCT, PROCES, TOP, and RM. Also it sets up arrays and zeros out memory space.

Variable Definition

Variables are defined in each subroutine where they appear.

Listed here are only the variables specifically referred to in this program.

NFLAG	- a flag to determine if the RM routine is to be called
MPARM	- a number to dimension arrays--the number of pollutant parameters
TEST	- a flag to indicate insufficient data for TOP
MSORS	- a number to dimension arrays--the number of sources
MBRNC	- a number to dimension arrays--the number of branches
MEQ	- a number to dimension arrays--the number of pieces of equipment
MBRP1	- the number of branches plus 1
MP2	- the number of branches plus 2
MASTER	- the largest parameter number to be used in the current problem
EPSLON	- a flag to determine if CORRCT is to be called

Statements

1	defines COMMON
2-7	dimensions all arrays in the program. These dimensions must be in agreement with numbers assigned MPARM, MSORS, MBRNC, MEQ, MBRP1, and MP2.
8-13	defines size of MPARM, MSORS, MBRNC, MEQ, MBRP1, MP2
14-40	zeros out arrays NSET, PCRM, PMDATA, SAMFRE, NMA, NALOW, Y, PC, PM, AMAR, NPLA, SMEQTI, EQTIME, CONSTR, SUMM.

41 MASTER is read in
42 NFLAG is read in
43 EPSLON is read in
44 format for reading MASTER, NFLAG, EPSLON!
45 CALL PROCES subroutine
46 CALL START subroutine
47 CALL TOP subrouting
48 if TEST = 1 terminate program if ALK and ACV are not included
in NPLIST
49 if EPSLON = 0 do not call CORRCT
50 Call CORRCT
51 if NFLAG = 0 call LEVEL and RM
52 Call LEVEL
53 Call RM
54 STOP
55 END

SUBROUTINE PROCES - This program accesses a library of subroutines each of which defines a process in terms of flow at 100% capacity operation and concentration or other measure number of pollutant parameter from the process. It forms two arrays, one for flows from each process and another for concentrations of each pollutant from each process. It also allows the modification of these arrays by changing individual parameters within the arrays or by replacing entire rows which represents an entire process.

Variable Definition

XNAME (MSORS)	- process names--read in when defining list of processes in a plant
YNAME (MSORS)	- process names--read in when performing modifications-- defines row in which modification is to be made
ELE (MPARM)	- parameter (element) in row to be modified
VAL (MPARM)	- value to which above parameter (element) is to be modified
ZNAME (25)	- process model names--name of process models which do not reside in library which user wishes to define
FLW (25)	- value which represents 100% flow of a new process model
FLOW (MSORS)	- array which holds values of process flows at full operating capacity
SPLIST (MSORS, MPARM)	- array which holds values of pollutant parameters-- rows relate to process model names--columns relate to each different pollutant name
CAP (MSORS)	- operating capacity of a process

Lines

1	Name of subroutine PROCES
2	Dimensions the arrays, XNAME, ELE, YNAME, VAL, ZNAME, FLW, FLOW, SPLIST, CAP
3	Sets the variable ELE to an integer variable
4- 54	Initialize the names of processes in the library
55- 60	Initialize the FLOW and SPLIST arrays to zero
61- 62	Reads N - the number of processes to be used in a particular run
63- 68	Reads and writes the names of each process to be used in a run and the corresponding operating capacity of each process.
69	Resets JM = 0
70-122	Selects and calls the appropriate subroutine for defining each process used in a particular run. These subroutines are described below
123-124	Reads NOD - this variable if "1" indicates that these are modifications to be performed on the SPLIST array - if "0" indicates that there are no modifications
125	Writes NOD
126	Checks value of NOD - if NOD = "1" program execution is transferred to statement 127 - if NOD = "0" program execution is transferred to Statement 140
127-128	Reads NMOD - the number of modifications to the SPLIST array
129	Writes NMOD
130-134	Reads and writes YNAME, ELE, and VAL variables
135-139	Compares YNAME's to XNAME's. When YNAME matches XNAME the value of the SPLIST element defined by JM and ELE(NM) is replaced by VAL(NM)

140-142 Reads and writes NPRO - if NPRO = "1" indicates that there are processes named in the XNAME array which are not in the library and which are to be defined now. If NPRO = "0" indicated that there are no undefined processes in the XNAME array.

143 Checks the value of NPRO - if NPRO = "1" execution is transferred to Statement 144. If NPRO = "0" execution is transferred to Statement 160.

144-145 Reads NUMP - the number of new processes to be defined now

146 Reads NELE - the number of pollutant elements to be placed in the SPLIST row pertaining to each new process being defined now.

147-159 Does the following:

1. Reads ZNAME - name of new process previously put in XNAME array and also reads FLW which is flow from new process at 100% capacity.
2. Compares ZNAME to XNAME array entries. When proper entry found this process is defined:
 - a. FLOW - flow for new process at operating capacity
 - b. SPLIST - elements are read in corresponding to new process.

This process is repeated for each new process until all new processes are defined.

160 Return statement - returns program execution to main program

161 END statement

PROCESS LIBRARY - Subroutines for defining processes in library.

One subroutine for each process in library.

Line

1 Subroutine name
2 DIMENSIONS FLOW, SPLIST, CAP
3 Defines flow of process at operating capacity
Flow (JM) = .01 * CAP (JM) in % * MGD at 100%

Next N The next N statements define the row of the SPLIST
statements array which correspond to this process. N being
the number of pollutants.
The next statement returns execution to the PROCESS
subroutine.
END statement

SUBROUTINE START - This routine reads in the number of sources, branches, parameters, the topology matrix, and selects the parameters to be used from the master list.

Variable Definition

A (MSORS, MBRNC)	- contains the topology matrix of up to 25 sources and 50 branches
NS	- number of sources
NB	- number of branches
NP	- number of parameters
NTOP	- number of outfalls from system
C (MSORS, MPARM)	- contains parameter concentration for each of 25 possible sources for up to 25 parameters. Selected from SPLIST
SPLIST (MSORS, MPARM)	- master parameter concentration list passed from the Process Model
NPLIST (MPARM)	- parameter equivalence array indicating the number of the parameter of the master list that is associated with the parameters being used
NBRNCH (MBRNC,2)	- contains the names (up to 8 characters) of up to 50 possible branches

Statements

1- 8	subroutine definition and common area
9-10	reads in number of sources, branches, parameters, and outfalls (NS, NB, NP, NTOP, respectively)
11-12	reads in the numbers of the parameters from the master list

13-16 reads in the topology matrix
17-19 reads in the names of the branches
20-26 selects the appropriate parameter concentrations from
 SPLIST which contains the information for the master
 list of parameters and compresses it into array C
27-32 initializes variables for COMMON, calls CHK1 and returns
 and ends

SUBROUTINE CHK1 - This routine reads in parameter names, sample frequencies, number of methods available for analysis, length of survey, and outputs topology matrix, and source parameter information

Variable Definition

A (MSORS, MBRNC)	- contains the topology matrix of up to 25 sources and 50 branches
NS	- number of sources
NP	- number of parameters
NB	- number of branches
C (MSORS, MBRNC)	- contains parameters concentrations for up to 25 sources
FLOW (MSORS)	- contains flow quantity from each source
LENGTH	- length of the survey (days)
POLN (MPARM,5)	- contains the names of the parameters (up to 20 characters)
ISTOP	- the number of non-competing parameters
XNAME (MSORS)	- contains the names of each source (up to 4 characters)
NTEMP (MPARM)	- contains the number of methods available to analyze each parameter
NPLIST (MPARM)	- parameter equivalence array
SAMFRE (MPARM)	- contains sample frequency for each parameter
NMA (MPARM)	- contains number of methods available to analyze each parameter (from master list)

Statements

1- 9 subroutine definition and COMMON area

10-11 reads in length of the survey
 12-13 reads in the number of non-competing parameters
 14-17 FORMAT statements for output, and page eject
 18-22 prints number of parameters, number of non-competing
 parameters, length of the survey, and labels
 23-33 reads in all parameter names, number of methods and
 sample frequencies from the mater list and saves only
 those that are currently being used. Also the total
 numbers of samples for the survey are calculated based
 on sample frequency and length of the survey
 34-37 prints labels and number of sources, branches, and
 outfalls
 38-41 prints the numbers of the sources, the names of the
 sources, and the rows of the topology matrix
 42-46 prints labels and the flow associated with each source.
 47-49 initializes counters and flag (for use if more than 6
 parameters are used)
 50 if more than 6 parameters are present go to 30 (statement 60)
 51-57 prints out parameter name, source name, and the corre-
 sponding concentrations for up to 6 parameters at all
 sources
 58-59 if more than 6 parameters go to 31 (statement 71), other-
 wise go to the return segment.
 60-61 there are more than 6 parameters so initialize counters
 the first time this segment is used
 62 if this segment has been used before, go to 32 (statement
 65)

63-64 increment counter and transfer to 33 (statement 51)
65 set I to the next parameter number
66-69 determines the upper limit on the parameter number (NNP).
 This segment is used to guarantee that no more than 6
 columns of parameters are printed at a time
70 transfers to print segment (statements 51-53)
71-72 if all parameters have been printed, return, otherwise,
 go to 30 (statement 60) to select next 6 parameters
73-75 CONTINUE, RETURN, AND END control statements

SUBROUTINE TOP - This subroutine calculates the flows, concentration of parameters, mass per day of conservative parameters in all branches of the system. The effects of treatment are included. Also calls a subroutine which flags parameters to be measured at specified measure points. Also calls CHK2 for read out of pertinent information.

Variable Definitions

A (MSORS, MBRNC)	- contains the topology matrix of up to MSORS sources and MBRNC branches
NS, NSS	- number of sources in a particular study
NP, NPP	- number of parameters in a particular study
NB, NBB	- number of branches in a particular study
C (MSORS, MPARM)	- contains the parameter concentrations for NS sources and NP parameters
Y (MBRNC, MPARM)	- output matrix of parameter concentration for each of NB branches and NP parameters.
FLOW (MSORS)	- contains the quantity of flow for each source (MGD)
X (MBRNC, MPARM)	- contains mass parameter for each of the NB branches
EFF (MBRNC)	- contains the flow for each branch
P (MSORS, MPARM)	- contains the mass parameter for each source {C (J, K) * FLOW (J)}
NPLIST (MPARM)	- lists the numbers (from the master list) of the parameters in this study
PT(25)	- branch parameters passed to treatment subroutines

EFFT	- branch flow passed to treatment subroutines
MSORS	- the program dimension for the number of sources
MPARM	- the program dimension for the number of parameters
MBRNC	- the program dimension for the number of branches
TEST	- a flag that is set if all source flows are not given
CRFLAG	- a flag that is set if TOP is called from CORRCT so that read in of treatment data is omitted
RANK	- variable for summing masses in FLAG subroutine
FLGPT	- flagging level for each parameter

Statements

1 - 7	subroutine definition and dimensions
8	set test flag = 0 Test = 1 informs MAIN that there is insufficient flow information to con- tinue
9 - 11	zero out X(I,J) array
12	flag to skip read in if TOP is called from CORRCT
13 - 28	read in treatment data Settle: AREA NEUTRA: MNEUT, MENU, NCAUS, MACY, DOME: ITYPE, TEMP, AREAP, AREAS, ABVOL, TFAREA, TFVOL, MLSS, YN, R1, R2, R3, R4, K20, NN
29 - 30	read out AREA
33 - 35	change pH to $\{H^+\}$
36 - 38	concentration C(I,J) times FLOW to give P(I,J) for each source
39 - 49	Do loop to compute the flow of each branch, EFF(I). If flow of any source is not given, print message and return to MAIN with TEST = 1.
50 - 51	set flags NAC and NALK = 0. These will indicate when alkalinity and acidity are not in NPLIST.
52 - 53	set constants XK1 and XK2
54 - 56	determine numbers for NAC and NALK
57 - 61	If both NALK and NAC are in NPLIST continue. If not print out message set Test = 1 and return.
62 - 142	calculates the mass parameter, X(I,K), in branch 1.

63 - 66 set variables and counters to zero
 67 start search through topology matrix to determine
 branch inputs
 68 if source J does not contribute to branch I, look
 at next source
 69 if current branch is not the first branch, go
 to 180 (statement 89)
 70 - 88 for the first branch calculate the contribution
 to $X(I,K)$ for each contributing source J for each
 parameter K. As each source J is added, alkalinity,
 acidity, and capacity factors CT are computed as
 conservative parameters.
 72 Source $\{H^+\}$ for computing CT is computed
 73 - 75 α_1 , α_2 , and α_0 are computed for source J
 76 - 79 if source alkalinity is known CT and acidity are
 computed from alkalinity
 80 for branch 1 look at next source
 81,87 if acidity is given for source J then CT and
 alkalinity are computed in terms of acidity.
 88 for branch 1 look at next source.
 89 sets II to one less the current branch number
 90 begins search back through columns of topology
 matrix
 91 L is the current column being checked
 92 - 93 if source J appears in branch L go to 200
 (statement 95) otherwise look at next branch back

94 if source has not been included in some previous
branch go back to 130 {statement 70} and compute
the contribution to X (I,K), including alkalinity,
acidity, and CT.

95 - 101 if source has been included in some previous
branch L, add the X (L,K) of that branch to the
current X (I,K) calculation unless that X (L,K)
has been included in X (I,K) when looking at a
previous source.

96 - 99 if the amount from branch L has already been
added to current branch, go to 230 (statement 104)
otherwise increment NKNT the number of branches
that have been added, and add to the list
{NTB (NKNT) = L}

102 - 103 compute CT1 for branch L and get composite CT

104 look at next source

105 - 123 given the conservative values of CT and
alkalinity for a mixture of flows in branch I,
{H⁺} is computed by iteration using flow weighted
{H⁺} as starting value.

105 approximate value of {H⁺} for starting

106 molar value for alkalinity

107 Composite CT with flow divided out

108 alkalinity from CT and approximate {H⁺}

109 - 111 test for Convergence

112 decrement pH by 0.1

113 - 117 recalculate ALK and test for convergence

118 increment pH by 0.1

119 - 123	recalculates ALK and test for convergence
124	this gives buffered pH for composite flow of branch
125 - 127	checks to see if the branch flow undergoes a new treatment in this branch. Note {the X(L,K) values previously added were treated values if appropriate} if a treatment occurs here go to 280 (statement 128) else go to next branch.
128 - 131	prepare variable for treatment subroutine
132	if A(J,I) = 2, then call NEUTRA
133	if A(J,I) = 3, then call SETTLE
134	if A(J,I) = 4, then call DOME
135 - 139	calling statements
140 - 141	restoring X(I,K) after treatment
142	end of loop for each branch
143 - 148	change mass variables back to concentrations {Y(I,J)} for output.
149 - 153	change the {H ⁺ } values to pH
159 - 157	change A matrix to - or 1 entries for later use
158 - 159	set NBB and NPP
160	call CHKZ for read out
161	call FLAG subroutine
162 - 163	RETURN and END statements

SUBROUTINE NEUTRA - This subroutine models the neutralization treatments.

It models acid or caustic neutralization and for each allows one of several neutralizing agents. The following flags are required and are read in in TOP subroutine

MNEUT = 1	acid neutralization
MNEUT = 2	caustic neutralization
MENU = 1	Ca(OH)_2
MENU = 2	CaCO_3
MENU = 3	Na_2CO_3
MENU = 4	NaOH
NCAUS = 1	H_2SO_4 or HCl or HNO_3
NCAUS = 2	CO_2
MACY = 1	H_2SO_4
MACY = 2	HCl
MACY = 3	HNO_3

Variable Definitions

NPLIST (MPARM)	- lists the numbers from the master list of the parameters in this study
PT (MPARM)	- branch parameters passed to treatment subroutine
EFFT	- branch flow passed to treatment subroutines
MPARM	- the program dimension for the number of parameters
NP	- the number of parameters in a particular study

Statements

- 1- 5 call and dimension statements
- 6, 7 constants for capacity factor (CT) calculation

8 PT(1) to H^+
 9-11 $\alpha_0, \alpha_1, \alpha_2$ for branch H^+
 12 if caustic neutralization go to 180 (statement 42),
 for acid neutralization continue
 13-19 compute CT in and ACYIN with either ALK_{in} or ACY_{in}
 given
 20-22 α_0 and α_1 for $H^+ = 7$ after treatment
 23 if acid neutralization treatment is by $Ca(OH)_2$ or
 NaOH go to 104 (statement 25)
 24 if acid neutralization treatment is by $CaCO_3$ or Na_2CO_3
 go to 105 (statement 29)
 25 CT_{out} for $Ca(OH)_2$ or NaOH
 (104)
 26 ACY_{out} for $Ca(OH)_2$ or NaOH
 27 if $Ca(OH)_2$ go to 106 (statement 33)
 28 if NaOH go to 107 (statement 79)
 29 ACY_{out} for $CaCO_3$ or Na_2CO_3
 (105)
 30 CT_{out} for $CaCO_3$ or Na_2CO_3
 31 if $CaCO_3$ go to 108 (statement 88)
 32 if Na_2CO_3 go to 209 (statement 133)
 33 $Ca(OH)$ added to neutralize to $pH = 7$
 (106)
 34 20% excess added
 35-96 do loop to determine if both Ca and SO_4 are included
 in study. If not print message and return.
 47 SO_4 in molar units
 48 total Ca in molar units
 49 active Ca in molar units

50-52 H^+ , ALK, ACY out after neutralization (all multiplied by
 flow)
 53 calculate solubility product (SOLYP)
 54 if $SOLYP \leq 1.32 \times 10^{-4}$ go to 20 (statement 69)
 55-57 calculate $CaSO_4$ precipitate
 58 remaining SO_4
 59 remaining Ca
 60-68 determine dissolved solids, total solids, suspended solids,
 and hardness return
 69-78 determine Ca, dissolved solids, total solids, suspended
 (20) solids and hardness if there is no precipitate. return
 79 Calculate NaOH required for neutralization
 (107)
 80-87 determine dissolved solids, total solids, suspended solids,
 pH, ALK, ACY. return
 88
 (108) calculate $CaCO_3$ required for neutralization
 89 20% extra added
 90-101 do loop to determine if both Ca and SO_4 are included in
 study. If not print message and return.
 102 $H^+ * EFFT$
 (112)
 103- ALK and ACY after neutralization (times flow)
 104
 105 SO_4 in molar units
 106 active Ca in molar units
 107 solubility product SOLYP
 108 $SOLYP \leq 1.32 \times 10^{-4}$ go to 113 (statement 123)
 109- determine $CaSO_4$ precipitate
 111

112 remaining SO_4
 113 remaining Ca
 114- determine dissolved solids, total solids, suspended solids
 122 suspended solids, and hardness. return
 123- if no precipitate determine Ca, dissolved solids, total
 132 solids, suspended solids and hardness. retur
 (113)
 133 determine Na_2CO_3 for neutralization
 (209)
 134- determine H^+ , ALK, ACY, dissolved solids, total solids,
 141 and suspended solids after neutralization. return
 142- for caustic neutralization - determine CT_{in} and ALK_{in}
 148 if either ALK_{in} or ACY_{in} is given
 (180)
 149- H^+ after neutralization, α_1 , and α_2 for pH = 7
 151
 152 if CO_2 neutralization go to 127 (statement 178)
 153- CT_{out} , ALK_{out} , H^+ , ACY after neutralization
 157
 158 if not H_2SO_4 neutralization go to 123 (statement 165)
 159 H_2SO_4 required for neutralization
 160- dissolved solids, total solids and SO_4 after neutralization.
 164 return
 165 if not HCl neutralization go to 125 (statement 172)
 (123)
 166 HCl required for neutralization
 167- dissolved solids, total solids, and chlorides after
 171 neutralization. return
 172 HNO_3 required for neutralization
 (125)
 173- dissolved solids, Nitrate/Nitrite, total solids after
 177 neutralization. return

178 ALKout for CO₂ neutralization
(127)

179 CTout for CO₂ neutralization

180 CO₂ required for neutralization

181- H⁺, ALK, ACY, dissolved solids, and suspended solids after
186 neutralization

187 return

SUBROUTINE SETTLE - This subroutine models the settling pond or clarifier. The clarifier area is read in acres TOP. SETTLE is called from TOP when a 3 appears in A(J,I).

Variable Definitions

PT(MPARM)	- branch parameters passed to treatment subroutine
NPLIST (MPARM)	- lists the numbers (from the master list) of the parameters in this study
EFFT	- branch flow passed to the treatment subroutine
MPARM	- the program dimension for the number of parameters
NP	- the number of parameters in a particular study

Statements

1-4	- call and dimension statements
5-6	- compute the removal factor (RF)
7	- starts DO loop for Settling solids
8	settles COD which is 30% dissolved
9	settles TOC which is 30% dissolved
10	if KJELDAHL N is in study go to 15 (statement 16)
11	if total solids is in study go to 25 (statement 22)
12	computes suspended solids after settling
13	computes vol. susp. solids after settling
14	computes turbidity after settling
15	go to 10 for K > 23 (statement 45)
16-18 (15)	go to 17 if ammonia is included (statement 20)
19	go to 10 if ammonia is not included (statement
20 (17)	compute settled Kjeldahl N

21 go to 10 (next K) (statement 45)
22-44 calculates total solids if any two of the following three
(25) are given: total solids, dissolved solids, suspended solids
45 next K
(10)
46-47 Return - end

SUBROUTINE DOME - This subroutine models the domestic or sanitary sewage treatment. This model includes a trickling filter with recycle or an activated sludge type treatment. DOME is called from TOP. Treatment system data is read in in TOP and includes:

TRICKLING FILTER

AREAP primary clarifies area in acre
ITYPE = 0 trickling filter
 = 1 activated sludge
TFVOL trickling filter volume in acre-feet
TEMP wastewater temperature ($^{\circ}\text{C}$)
R2 recycle from filter effluent (MGD)
R3 recycle from secondary clarifier (MGD)
R4 recycle to the filter (MGD)
TFAREA trickling filter area in acres
K20 = 0.23 for 1" rock media
 = 0.13 for 2 1/2" rock media
AREAS secondary clarifies area in acres

ACTIVATED SLUDGE

AREAP as above
ITYPE as above
AREAS as above
ABVOL aeration basin volume in MG
MLSS mixed liquor suspended solids (Mg/l)
YN net yield computed from plant log data
R1 activated sludge recycle (MGD)

Variable Definitions

- NPLIST (MPARM) - lists the numbers (from the master list) of the parameters in this study
- PT (MPARM) - branch parameters passed to the treatment subroutine
- EFFT - branch flow passed to treatment subroutine
- MPARM - the program dimension for the number of parameters
- NP - number of parameters in a particular study

Statement

- 3- 6 Initialize data
- 7- 8 Store input water quality parameters in array TT
- 9 Calculate removal factor for primary clarification
- 10-12 Calculate primary clarifier effluent total solids
- 13 Calculate primary effluent TKN
- 14 Calculate primary effluent TSS
- 15 Calculate primary effluent VSS
- 16 Calculate primary effluent Turbidity
- 17 Calculate primary effluent COD
- 18 Calculate primary effluent TOC
- 19 Calculate primary effluent BOD
- 20 Route to 3 for trickling filter system or 4 for activated sludge system
- 21 Calculate recirculation ratio
- 22 Calculate organic loading
- 23 Calculate recirculation factor
- 24 Calculate filter BOD removal factor

- 25 Correct BOD removal factor for water temperature
- 26 Calculate filter effluent suspended solids
- 27 Calculate filter effluent COD
- 28 Calculate filter effluent TOC
- 29 Calculate filter effluent BOD
- 30 Calculate depth of filter
- 31 Calculate hydraulic loading on filter
- 32 Correct media coefficient for water temperature
- 33 Calculate filter nitrification factor
- 34 Calculate filter effluent nitrates
- 35 Calculate filter effluent TKN
- 36 Calculate filter effluent ammonia
- 37 Calculate flow to secondary clarifier
- 38 Calculate removal factor for secondary clarifier
- 39 Calculate secondary effluent suspended solids
- 42 Calculate process loading intensity
- 43 Calculate sludge age
- 44 Calculate flow to aeration basin
- 45 Calculate aeration basin effluent BOD
- 46 Calculate aeration basin effluent COD
- 47 Calculate aeration basin effluent TOC
- 49 Calculate aeration basin effluent suspended solids
- 50 Change area of secondary from acres to square meters
- 51 Calculate secondary solids surface feed
- 52 Calculate secondary effluent suspended solids
- 53 Check for nitrification requirement
- 54-57 Initialize coefficients
- 58 Calculate aeration basin detention time

59 Make initial guess of Nitrosomonas concentration
60-61 Set Nitrosomonas limits
62 Set iteration counter for Nitrosomonas
63 Make initial guess of ammonia concentration
64-65 Set ammonia limits
66 Set iteration counter for ammonia
67-70 Make calculations on ammonia and Nitrosomonas
71-82 Test ammonia concentrations, adjust it necessary and
 repeat steps
99 Calculate aeration basin effluent nitrates
100 Calculate aeration basin effluent TKN
101 Calculate aeration basin effluent ammonia
103- Store changed parameters in array PT
104

SUBROUTINE FLAG - This subroutine flags parameters for measurement at a source based on fraction of total mass contribution, and also flags any parameters at sources and branches selected by the user.

Variable Definition

FLGPT (MPARM)	- an array containing the fractional level of total mass for which a parameter will be flagged at a source (the first branch in which a source appears)
POLN (MPARM, 5)	- an array containing the names of the parameters
NP	- number parameters
NS	- number of sources
NB	- number of branches
RANK (MPARM)	- contains total masses of parameters
P (MSORS, MPARM)	- contains mass of each parameter at each source
A (MSORS, MBRNC)	- topology matrix (up to 25 sources and 50 branches)
Y (MBRNC, MPARM)	- contains the concentration in each branch (50 possible) of each parameter
NF	- number of flagged sources (selected by user)
NX (50)	- contains the source number (or branch number) for
NY (50)	- contains the parameter number for a selected flag point. There is a one to one correspondence between NX and NY (eg. NX (3) contains the source (or branch) number for flagged point 3, and NY (3) contains the parameter number for flagged point 3)
NFB	- number of flagged branches (selected by user)
XNAME (MSORS)	- contains the name for each source

Statements

- 1- 8 subroutine definition, COMMON and DIMENSION area
- 9-10 reads into FLGPT the minimum mass contribution flagging levels for the parameters (from the list chosen by the user)
- 11-15 prints out a title followed by the name of each parameter and its flagging level
- 16-18 adds up the mass for each parameter from each source and puts the total in RANK
- 19-28 calculates the fraction of mass contributed by each source for each parameter (TEMP), and if this is greater than or equal to the flagging level, the topology matrix (A) is searched to find the first branch where the source appears, and then the corresponding concentration in the branch-parameter concentration matrix (Y) is flagged by setting it negative
- 29-31 the number of flagged sources (selected by the user) is read in, and if there aren't any, go to 80 (statement 51)
- 32-34 each source (NX) and the parameter for that source (NY) are read in
- 35-41 a title is printed followed by the name of each source and parameter flagged
- 42-50 searches the topology matrix to find the first branch where a source appears, then sets the corresponding entry in the branch-parameter concentration matrix negative to indicate the flag

51-53 reads in the number of flagged branches (selected by the
 user) and if this is zero, the routine is ended

54-64 prints a label followed by the name of each branch and
 parameter flagged, then sets the appropriate entry in
 the branch-parameter concentration matrix (Y) negative
 to indicate the flag for each of the flagged points then
 returns

SUBROUTINE CHK2 - This subroutine outputs parameter concentration and
mass information

Variable Definition

Y (MBRNC, MPARM)	- output matrix of parameter concentration for each of 50 possible branches
NB	- number of branches
NP	- number of parameters
NBRNCH (MBRNC, 2)	- name of each branch (up to 8 characters) for up to 50 branches
POLIN (MPARM, 5)	- name of each parameter (up to 20 characters) for up to 25 parameters
X (MBRNC, MPARM)	- output matrix of parameter mass for each of 50 possible branches
EFF (MBRNC)	- flow in each of 50 possible branches

Statements

1- 5	subroutine definition, common area, and dimensioning
6	sets flag to use in determining which output segment is being done (concentration or mass)
7-11	prints title on page and outputs name of branch and flow in that branch
12-15	prints title on page and outputs number of branches and parameters
16-18	initializes counters and flag (for use if more than 6 parameters are present)
19	if more than 6 parameters are present go to 30 (statement 29)
20-26	prints out parameter name, branch name, and the corresponding concentrations for up to 6 parameters at all branches

27 if more than 6 parameters are present go to 31 (statement 40)
28 go to 5 (statement 42)
29-30 there are more than 6 parameters, so initialize counters
 the first time this segment is used
31 if this segment has been used previously go to 32
 (statement 34)
32-33 increment counters and transfer to 33 (statement 20)
34 set I to the next parameter number
35-38 determines the upper limit on the parameter number
 (NNP). This segment is used to guarantee that no more
 than 6 columns of parameters are printed at a time.
39 transfers to print segment (statement 20-26)
40-41 if all parameters have been printed go to 5 (statement 42),
 otherwise, go to 30 (statement 29) to select next parameters
42-43 if both concentrations and masses have been printed go to
 60 (statement 52)
44-49 prints title and converts mass to pounds per day (array
 Y is used temporarily for this information)
50-51 sets the flag and goes to 55 (statement 16)
52-57 restores concentrations in array Y
58-66 converts H^+ concentration to pH
67-68 RETURN and END control statements

SUBROUTINE LEVEL - This subroutine provides the interface between the Topological and Resource Models. It determines from the topology matrix all possible levels of the system, a level defined as a set of points (branches) which completely characterizes all outfalls of the system. Once the levels have been found the Y matrix is transformed to the P matrix.

Variable DEFN

SUM (MSORS)	- contains linear combinations of columns of the topology matrix
SUMA (MSORS)	- contains previous contents of SUM
SOL (SI, MBRPI)	- array containing the branch number for each level (IBN). SOL (1, 1) contains the number of points (branches) in level 1. SOL (1, 2-50) contains the corresponding branch numbers for each point in the level
TOP (100)	- contains sequences of branches which are being treated as solution sequences
SUMRR	- used to sum the rows of the topology matrix
Y (MBRNC, MPARM)	- output matrix of parameter concentration for each of 50 possible branches
NP	- number of parameters
NS	- number of sources
NB	- number of branches
NLEV	- number of levels
NTOP	- number of outfalls
A (MSORS, MBRNC)	- topology matrix

29 Tests the value of K. If K equals zero a solution sequence has been found and control transfers to 7 (statement 41). If K is equal to one the trial sequence has not violated the solution sequence, but another branch must be added to the sequence.

30-31 stores the new SUM (J) totals on the SUMA array

32-35 adds one more element on trial solution and sets M to the new branch to be added equal to one less than the last branch. If M is less than one, go to 47 (statement 94), otherwise, go to 6 (statement 22) and test the new sequence.

36-37 replace SUM with SUMA (the previous values of SUM)

38-40 sets new trial branch to be added to trial sequence to old trial branch minus 1. If new trial branch is greater than or equal to 1 control is transferred to 6 (statement 22) and the new sequence is tested. Otherwise, go to 47 (statement 94)

41 increments the number of solutions found (JJ)

42 stores in row JJ column 1 of SOL the number of branches in the new solution sequences

43-44 stores in row JJ column 2, $1 + 1$ the branch numbers of the new solution sequences

45 if the maximum number of solution sequences (levels) has been found go to 20 (statement 141)

46 if the last branch in the new solution sequence is not equal to one, go to 5 (statement 36) and the case is handled as if the last element in the solution sequences had been a violation

47 if only one solution sequence has been found, go to 55
 (statement 78)
 48 set KK to zero
 49-53 determine the position (KK) of the last common entry
 in the last two solution sequences found
 54 if there are no common entries go to 55 (statement 78)
 55 decrements the branch in the $KK + 2$ position of the
 solution sequence by 1.
 56 if IR is less than or equal to one go to 351 (statement 74)
 57-58 set the next branch in the trial sequence (M) to IR and
 set the number of branches in the trial sequence to KK
 59-60 transfer the first KK elements in the last solution
 sequence into positions 1 to KK-1 of TOP
 61 sets LIMIT to KK (number of branches in trial sequence)
 62-64 resets SUMA and SUM arrays to zero
 65-70 store the sum of the columns of topology matrix
 designated in TOP into SUMA and SUM arrays
 71 go to 6 (statement 22)
 72-73 if KK (number of branches in trial sequence) is zero,
 go to 300 (statement 128) else go to 271 (statement 55)
 74-75 decrement the number of branches in the trial sequence
 and go to 362 (statement 72)
 76-77 set the number of branches in the trial sequence to
 one and go to 271 (statement 55)
 78-82 determine KK when only one solution sequence has been
 found, and go to 300 (statement 128)

83 if only one branch is in the trial sequence, go to 300
 (statement 128)
 84-85 determine the last common entry and if it is less than
 one, go to 651 (statement 92)
 86-87 set M (the next branch in the trial sequence) to IR, and
 the number of branches in the trial sequence to KK
 88-89 set TOP to the last solution sequence
 90-91 set LIMIT2 to the number of branches minus one and go
 to 69 (statement 62)
 92-93 decrement the number of branches and go to 970
 (statement 83)
 94-97 if no solution sequences are found print an error
 message and return, otherwise, go to 360 (statement 98)
 98-103 determine KK as the position of the last common entry
 between the trial solution sequence whose last branch
 is one and the last solution sequence found.
 104 if there is no common entry, go to 855 (statement 114)
 105-106 if the common branch is less than or equal to one, go
 to 851 (statement 110)
 107-109 save the common branch number, the number of branches
 110-111 decrement the number of branches and go to the next
 statement
 112-113 if there are no branches, go to 300 (statement 128),
 otherwise, go to 871 (statement 105)
 114-118 determines KK (number of branches in solution sequence)
 from TOP
 119 if there is only one branch, go to 300 (statement 128)

120-121 if the common branch is one or less, go to 951 (statement 126)
 122-123 saves the common branch and the location of the newly added
 branch
 124-125 sets LIMT2 to the previous position and go to 69
 (statement 62)
 126-127 decrement KK and go to 670 (statement 119)
 128-134 check to see if any more solution sequences exist. If
 not, go to 20 (statement 141)
 135-136 try a new sequence whose first element (M) equals the
 first branch in the TOP array minus one and set I to one.
 137-140 reset SUMA and SUM arrays to zero and go to 6 (statement 22)
 141-145 determines NMAX and NEMAX as the maximum number of branches
 in a solution sequence (level)
 146-147 defines DO loop counters LIM2 and LIM3
 148-159 rearranges the rows of the SOL array in increasing
 number of branches in the levels
 160 tests to see if there is more than one outfall from
 the system. If so, go to 33 (statement 169)
 161 defines DO loop counter LIM6 (number of solution
 sequences)
 162-168 moves the levels in SOL down and inserts the level that
 consists of only the outfall, and goes to 52 (statement 170)
 169 redefines LIM6
 170-180 calculates P (I, J, K) array from the Y array and level
 information of SOL array

181-183 defines number of levels (NLEV), maximum number of branches
in a level (NPMAX), and the number of parameters (NOPL)
184-185 sets up NPLA array
186-187 RETURN and END control statements

SUBROUTINE RM - This subroutine reads in and prints out resource information, allocates resources and calls subroutines when needed to relieve violations, and prints out resource allocation information.

Variable Definition

NL	- number of levels
NP	- number of parameters
NPLA (25)	- number of points in each level
NPOFEQ	- number of pieces of equipment
EQNAME (MEQ, 5)	- names of equipment items
EQTIME (MEQ)	- time available for each item of equipment
NPLIST (MPARM)	- parameter equivalence array
NMA (MPARM)	- number of methods available for each parameter (in the current parameter list)
NTEMP (MPARM)	- number of methods available for each parameter in the master list that are not in the current list
MENAME (MPARM,3,5)	- the names of the methods
PCRM (MPARM, 3)	- the minimum acceptable concentration for each method
PMDATA (MPARM, 3, 4, 5)	- contains resource information for up to 3 methods for each parameter. The information is stored as follows: first column - equipment item codes (up to 4 per method), second column - equipment times per sample, third column - van space requirement for each item, fourth column - analysts' times, fifth column - cost per sample, set up time, location of analysis, and time constraint

VNSP (MEQ) - van space requirement for each item of equipment

POLN (MPARM,5) - names of each parameter

CNSTAR (6) - constraints on van space, analysts' times, and cost

IDO (MPARM, 25) - if for a parameter I at level J all points are
flagged, then IDO (I, J) = 1, otherwise, it is zero

PM (MPARM,25,3) - array containing feasible method numbers (up to 3)
for each parameter at each level. If an entry is
zero, then no method is feasible at that level as a
whole (although a method might be feasible at a point
in a level).

AMAR (MPARM, MP2) - array containing selected methods. AMAR (I, 1)
contains the method number for parameter I. AMAR (I, 2)
contains the level number at which the method is used,
and AMAR (I, 3-25) contains the number of the method
at the corresponding point in the level if the
concentration is greater than zero.

IVIOL - violation code as follows: 1 - equipment violation,
2 - van space violation, 3 - 6 analyst violation
(analysts 1 - 4 respectively), 7 - cost violation

USENO (MEQ) - contains the number of times each item of equipment
is used

SMEQTI (MEQ) - sum of equipment time for each item

SUMM (6) - contains totals of van space, analysts' times, and
cost

Statements

1- 10 subroutine definition, COMMON and DIMENSION area

11 initializes the violation indicator

12- 16	prints out a lavel and the number of points at each level
17- 20	reads in and prints out the number of pieces of equipment
21- 26	reads in and prints out the equipment names and times available for each item
27	initializes the counter for NPLIST
28	D0 loop to scan the master list
29- 30	determines the number of methods available for each parameter
31- 32	D0 loop to read in all methods
33- 35	D0 loop to read in resource information (PMDATA)
36- 37	if the parameter under consideration is in the current list, increment N, then look at the next parameter
38- 39	FORMAT statements for reading
40- 73	D0 loop to look at all paramaters in the current list and to print out all information read in about them
74	call subroutine CONCK to determine feasible methods and make allocations for flagged points
75- 86	read in and print out constraints on van space, analysts' times, and cost
87- 89	initialize LVIOL and skip output to a new page
90	D0 loop to allocate for all parameters
91- 92	initialize LREM (initial method) and LT (flag indicating method assignment)
93	D0 loop to scan all levels
94	if all points at the current level are flagged, don't allocate any more resources and look at the next parameter

95	obtains the number of points in the current level
96	DO loop to scan all methods
97- 98	if there are no more methods at the current level to to the next level
99-102	if a feasible method is found on the first attempt, go to 2050 (statement 103), otherwise, set flag to two and save the first method and level number
103-105	save the method number in AMAR, decrement the flag (LT) and save the level number
106-114	call the appropriate subroutines to check for violations using the current method. If there are violations control is transferred to 2200 (statement 115). Otherwise, the allocations are added and allocations for the next para- meter are started.
115-118	if this is not the first time through this segment, go directly to another method, otherwise, save the first violation number to occur (and if it is, an equipment number) and then look at another method
119	look at another level
120-122	control reaches this point if no method at any level was found that did not relieve a violation. Resources are allocated for the present parameter by the first feasible method and level
123	if an equipment violation could not be relieved, and the violation is on the same equipment item as the initial violation, go to 4000 (statement 132)

124	if a violation could not be relieved and it's the same violation as the initial violation, go to the statements dependent on the type of violation
125-126	attempts to relieve violation on equipment. If successful go to 1000 (statement 149). If not go to 4000 (statement 132)
127-128	attempts to relieve violation on van space. If successful go to 1000 (statement 149). If not go to 4100 (statement 137).
129-130	attempts to relieve violation on analyst time. If successful go to 1000 (statement 149). If not go to 4200 (statement 141).
131	attempts to relieve violation on cost. If successful go to 1000 (statement 149). If not go to 4300 (statement 146).
132-136	print out violation information and save violation number and equipment number, and go to 1000 (statement 149)
137-140	print out violation information and save violation number, and go to 1000 (statement 149).
141-145	print out violation information and save violation number, and go to 1000 (statement 149)
146-149	print out violation information and save violation number, and look at the next parameter
150-151	calls subroutines to print out measure information and to allocate time for sampline and flow measurement
152	initializes flag point counter
153	DO loop to look at all parameters

154-159	calls subroutine to print out parameter allocation information
160-162	prints out a label and calls a subroutine to obtain equipment usage information
163	DO loop to look at all possible equipment items
164	if an item of equipment has not been used, go to 5200 (statement 175)
165-170	initializes variables and determines if an equipment time violation occurred, and the amount of time an item was used
171-175	calls subroutine to convert minutes to hours and minutes, and prints out equipment usage information, then looks at the next equipment item
176-182	prints out van space usage information (van space used, if a violation occurred, and the amount of the violation)
183-198	prints out the grand total of each analysts' time, of a violation occurred, and the amount of the violation
199-205	prints out the grand total of cost, if a violation occurred, and the amount of the violation
206-207	RETURN and END control statements

SUBROUTINE CONCK - This subroutine determines feasible methods at all levels for all parameters, and allocates resources to flagged points.

Variable Definition

NP	- number of parameters
POLN (MPARM, 5)	- names of parameters
MENAME (MPARM, 3, 5)	- names of methods (up to 3 per parameter)
FP (400, 4)	- flagged point information. FP (1, 1) contains the parameter number for flagged point 1. FP (1, 2) contains level number, FP (1, 3) contains point number, and FP (1, 4) contains the method number
IFPT	- number of flagged points
IDO (MPARM, 25)	- if for a parameter I, at a level J, all points are flagged, IDO (I, J) = 1, otherwise it is zero.
NL	- number of levels
NMA (MPARM)	- contains number of methods available to analyze
NPLA (25)	- contains the number of points at each level
PC (MPARM, 25, MPARM)	- contains the parameter concentration information. PC (I, J, K) contains the concentration of parameter I at point K of level J
PCRM (MPARM, 3)	- contains the minimum acceptable concentration for each method (3 possible) for each parameter
PM (MPARM, 25, 3)	- array containing feasible method numbers (up to 3) for each parameter at each level

IBN (51, MBRPI)	- array containing the branch numbers for each level. IBN (I, 1) contains the number of points (branches) in level I. IBN (I, 2-50) contains the corresponding branch numbers for each point in level I.
NALOW (MPARM, MBRNC)	- contains flags to determine if a parameter is measured at a branch. If NALOW (I, J) is a one then parameter I is measured at branch J, zero if it is not measured.
SMEQTI (MEQ)	- sum of equipment time for each item
SAMFRE (MPARM)	- the sample frequency of each parameter
EQUSED (MPARM, MEQ)	- number of times a piece of equipment has been used for a particular parameter
EU (MEQ)	- total number of times each item of equipment is used
SUMM (6)	- contains sums of van space, analysts' times, and cost for previous allocations
LENGTH	- length of the survey
NSET (MPARM, 3)	- array to indicate if set up time for a given method has already been added to the total
PMDATA (MPARM, 3, 4, 5)	- array containing resource information for up to 3 methods for each parameter. The information is stored as follows: first column - equipment codes (up to 4 items for each method), second column - equipment time per sample, third column - van space for each item, fourth column-

analysts' times, fifth column - cost per sample,
set up time, location of analysis, and time
constraint.

Statements

1- 6	subroutine definition, COMMON, and DIMENSION area
7	DO loop to look at all parameters
8	sets counter to zero (MNCK indicates if a feasible method has been found)
9	DO loop to look at all levels
10	sets method counter (MC) to zero
11	determines the number of methods available for the current parameter
12	DO loop to look at all methods
13-14	DO loop to check all points at the current level
15	if the concentration is zero, go to 400 (statement 17)
16	if the concentration is less than the minimum acceptable for the current method go to 300 (statement 21)
17	look at the next point in the level
18-20	increment MNCK to indicate a feasible method has been found, increment the method counter and store the number of feasible method
21	look at the next method
22	look at the next level
23	if no feasible method was found, allocate the first method anyway
24	look at the next parameter
25	initialize the flag point counter to zero

26-27 DO loops to look at all parameters and all levels
 28-30 initialize flagged point indicator array, flagged points
 at a level and feasible method indicator
 31-32 DO loop to look at all points in the current level
 33 if a point is not flagged go to 700 (statement 72)
 34 if a method is available go to 750 (statement 44)
 35-36 determine the number of methods available and look at
 all of them
 37-38 if the method is feasible go to 735 (statement 42),
 otherwise, look at the next method
 39 set LN to indicate no feasible method was found
 40-41 since no method was feasible use the first method anyway
 and go to 750 (statement 44)
 42-43 save the feasible method number and set LN to indicate
 a feasible method was found at a point
 44-46 determine the branch that the level and point correspond
 to, and if a measurement has already been made there, go
 to 1000 (statement 67), otherwise, set NALOW to indicate
 that a measurement will have been made
 47 obtain the method number
 48 DO loop to add all equipment times, van space, analysts'
 times and cost
 49-53 if no equipment item is used, go to 910 (statement 56),
 otherwise, add new equipment time to previous total, and
 increment the number of times the item is used

54-55	if the item has been used previously, go to 910 (statement 56), otherwise, add van space requirement to previous total
56-58	if set up time has already been added, go to 900 (statement 59), otherwise, add set up time to the appropriate analysts' total, then set the array to indicate that set up time has now been added
59-60	add sample time requirement to the analysts' total and close the loop (statement 48)
61	add new cost to the previous total
62-66	increment the flagged point counter and save the parameter number, level number, point number, and method number
67	if no feasible method existed go to 925 (statement 70)
68-69	if a feasible method existed only at a point, reset PM to zero and go to 940 (statement 71)
70	if the level is not the first level, reset PM to zero
71-72	increment the number of flagged points at a level and look at the next point
73	if all points at a level are flagged, set ID0 to one
74	look at the next level
75	look at the next parameter
76-77	RETURN and END control statements

SUBROUTINE T0 - This subroutine is used to zero out temporary allocations
for equipment times.

Variable Definition

TEMP (4, 2) - contains equipment number and total time allocated
for that item for the method under consideration
(up to 4 items per method)

Statements

1-2	subroutine definition and dimension statement
3-6	zeros out TEMP
7-8	return and end

SUBROUTINE EQCHEC - This subroutine determines if any equipment time violations occur for a method under consideration.

Variable Definition

I	- current parameter number
J	- current level number
L	- current method number
MC	- not used
NPLAJ	- violation code
IVIOL	- violation code
IEQN	- equipment number
PC (MPARM, 25, MBRNC)	- parameter concentration at a level and a point (eg. PC (2, 4, 7) is concentration of parameter 2 at point 7 of level 4)
AMAR (MPARM, MP2)	- array containing feasible methods. AMAR (1, 1) contains the method for parameter 1. AMAR (1, 2) contains the level that the method is used at, and AMAR (1, 3-25) contains the number of the method if it is used at the corresponding branch in the level
PMDATA (MPARM, 3,4,5)	- array containing resource information for up to 3 methods for each parameter. The information is stored as follows: first column - equipment codes (up to 4 items for each method), second column - equipment time per sample, third column - van space for each item, fourth column - analysts' time, fifth column - cost per sample, set up time, location of analysts' and time constraint
TEMP (4,2)	- contains equipment number and total time allocated for that item for the method under consideration

SAMFRE (MPARM) - sampling frequency for each parameter (for the total survey)

SMEQTI (MEQ) - sum of equipment time for each item

EQTIME (MEQ) - total time available for each item of equipment

Statements

- 1- 4 subroutine definition, COMMON and DIMENSION area
- 5 DO loop definition, to check all points at the level under consideration
- 6- 9 if parameter concentration at point K of level J is greater than zero, go to 150 (statement 11), otherwise set the corresponding branch entry in AMAR to zero and check the next branch
- 10 sets the corresponding branch entry in AMAR to the method under consideration
- 11-22 selects the number of the equipment item (if no equipment, IEQN = 0 so skip the time check), calculates the time used for the method under consideration, and checks if this will violate a time constraint. If it does then the error flag is set (IVIOL = 1) and the special return is used. Otherwise, a normal return is executed. This is done for each of four possible items of equipment.

SUBROUTINE VSCHEC - This subroutine checks for van space-constraint violations.

Variable Definition

I	- current parameter number
L	- current method number
IVIOL	- violation code
SUM	- van space used in current method
PMDATA (MPARM, 3, 4, 5)	- array containing resource information for up to 3 methods for each parameter. The information is stored as follows: first column - equipment item codes, second column - equipment time per sample, third column - van space for each item, fourth column - analysts' times per sample, fifth column - cost per sample, set up time, location of analysis, time constraint
EU (MEQ)	- number of times each item of equipment is used
SUMM (6)	- contains sums of van space, analysts' times, and cost for previous allocations
CNSTAR (6)	- contains constraints for van space, analysts' times, and cost

Statements

1- 3	subroutine definition, COMMON and DIMENSION area
4	sets current van space allocation to zero
5-11	computes van space needed for current method. If an item of equipment has been used previously (EU (IEQN) greater than 1), it is not added to the current van space

12-16 determines if a van space violation occurs. If not a
normal return is executed. Otherwise, the violation
code is set and a special return is executed.

SUBROUTINE CNCHEC - This subroutine determines if any analyst time violations or cost violations occur.

Variable Definition

ICONNO	- analyst number (or row number of cost information)
I	- current parameter number
J	- current level number
L	- current method number
ICP	- column number of analyst (or cost) information
IVIOL	- violation code
TOT	- total of an analysts' time (or cost) for current method
NPLA (25)	- number of points in each level
NSET (MPARM, 3)	- array to indicate if set up time for a given method has already been added to the total
LENGTH	- length of the survey
AMAR (MPARM, MP2)	- array containing possible methods. AMAR (1, 1) contains the method for parameter 1. AMAR (1, 2) contains the level that the method is used, and AMAR (1, 3-25) contains the number of the method at the corresponding point in the level if the concentration is greater than zero.
SAMFRE (MPARM)	- sampling frequency for each parameter
SUMM (6)	- contains sums of van space, analysts' times and cost for previous allocations
CNSTAR (6)	- contains constraints of van space, analysts' times and cost

PMDATA (MPARM, 3,4,5) - array containing resource information for
up to 3 methods for each parameter (see
variable definition for SUBROUTINE EQCHEC)

Statements

1- 4	subroutine definition, COMMON and DIMENSION area
5	sets current time total (or cost) to zero
6	sets NPLAJ to the number of branches in level J
7	if cost is being calculated go to 20 (statement 14)
8	if set up time has already been added for this method go to 20 (statement 14)
9	if analyst ICONNO is used for this method, calculate the set up time
10-13	for each branch of the level for which the current method is used calculate analysts' time (or cost) and add it to the total
14	if analysts' times are being considered go to 15 (statement 18)
15	if no cost was added, return without checking for a violation
16-17	if no cost violation return normally, otherwise, go to 50 (statement 21)
18	if no time was added, return
19-20	if no time violation return normally, otherwise, go to 50 (statement 21)
21-24	if a violation has occurred, set the proper violation code and execute a special return and END control statement

SUBROUTINE ADD - This subroutine adds the current allocations of equipment time, van space, analysts' times, and cost to the previous totals.

Variable Definition

- | | |
|--------------------------|--|
| I | - current parameter number |
| J | - current level number |
| L | - current method number |
| NPLA (25) | - number of branches in each level |
| PC (MPARM, 25,
MBRNC) | - parameter concentration at a level and a point
(eg. PC (2, 4, 7) is the concentration of parameter 2 at a point 7 of level 4) |
| PMDATA (MPARM, 3,4,5) | - array containing resource information for up to 3 methods for each parameter. The information is stored as follows: first column - equipment codes (up to 4 items for each method), second column - equipment time per sample, third column - van space for each item, fourth column - analysts' time, fifth column - cost per sample, set up time, location of analysis and time constraint |
| SMEQTI (MEQ) | - sum of equipment time for each item |
| SAMFRE (MPARM) | - sampling frequency for each parameter (for the total survey) |
| EQUSED (MPARM, MEQ) | - number of times a piece of equipment has been used for a particular parameter |
| EU (MEQ) | - total number of times each item of equipment is used |

- SUMM (6) - contains sums of van space, analysts' times,
and cost for previous allocations
- NSET (MPARM, 3) - array to indicate if set up time for a given
method has already been added to the total

Statements

- 1- 6 subroutine definition, COMMON and DIMENSION area
- 7 sets NPLAJ to the number of branches in level J
- 8 DO loop to look at each branch in level J
- 9 if parameter has zero concentration in a given branch do
not add anything to the totals; check the next branch
- 10 DO loop to look at all possible equipment items used
and analysts' used
- 11-12 if no equipment is used don't add any equipment usage
transfer to 300 (statement 18)
- 13 add the time to the appropriate equipment time total
- 14-15 increment the number of times a piece of equipment is used
- 16 if an item of equipment has been used before, go to 300
statement (statement 18) to avoid adding van space again
- 17 add van space requirement for equipment item
- 18 add analysts' time to total
- 19-20 if set up time has been allocated previously go to 200
(statement 21) to avoid adding it, otherwise, if an analyst
is used for this method, then add the set up time
- 21 end of DO loop
- 22 adds cost to total
- 23-25 end of DO loop and RETURN and END control statements

SUBROUTINE SET - This subroutine sets up the array of methods used at a branch in a level for a given parameter.

Variable Definition

- | | |
|--------------------------|--|
| I | - current parameter number |
| J | - current level number |
| L | - current method number |
| PC (MPARM, 25,
MBRNC) | - parameter concentration at a point and a level
(eg. PC (2, 4, 7) is the concentration of a
parameter 2 at point 7 of level 4) |
| A (MPARM, MP2) | - array containing possible methods (AMAR).
A (1, 1) contains the method for parameter 1.
A (1, 2) contains the level number that the
method is used, and A (1, 3-25) contains the
number of the method at the corresponding
point in the level if the concentration is
greater than zero. |
| N (25) | - number of points in each level (NPLA) |

Statements

- | | |
|------|--|
| 1- 3 | subroutine definition and DIMENSION area |
| 4 | sets A (1, 1) to the current method number |
| 5 | sets A (1, 2) to the current level number |
| 6 | sets NP to the number of points in level J |
| 7 | DO loop to check every point in level J |
| 8-10 | if the concentration at point K in level J is greater
than zero go to 40 (statement 11), otherwise, set A
(1, K + 2) to zero and check the next point. |

- 11-12 sets the corresponding point in array A to the current
method number and check the next *point*
- 13-14 RETURN and END control statements

SUBROUTINE PIEQCH - This subroutine searches past assignments to attempt
to alleviate constraint violations

Variable Definition

I	- current parameter number
ISTOP	- number of noncompeting parameters
IEQNR	- code of equipment items for which a violation has occurred
NOV	- violation code
AMAR (MPARM, MP2)	- array containing allocated methods. AMAR (1, 1) contains the method number for parameter 1. AMAR (1, 2) contains the level number at which the method is used, and AMAR (1, 3-25) contains the number of the method at the corresponding point in the level if the concentration is greater than zero.
PM (MPARM, 25, 3)	- array containing feasible method numbers (up to 3) for each parameter and each level
NPLA (25)	- number of points at each level
PMDATA (MPARM, 3, 4, 5)	- array containing resource information up to 3 methods per parameter. The information is stored as follows: first column - equipment codes (up to 4 items for each method), second column - equipment time per sample, third column - van space for each item, fourth column - analysts' times, fifth column - cost per sample, set up time, location of analysis, and time constraint
NL	- number of levels

Statements

- 1- 5 subroutine definition, COMMON and DIMENSION area
- 6- 7 prints out violation information
- 8 sets IMO to the current parameter number
- 9-10 looks at the next lower parameter and if it is a
noncompeting parameter, RETURN 1.
- 11-12 obtains the method number (LR) and the level number (JR)
- 13-14 saves the method and level numbers first tried
- 15-16 prints out method information
- 17 if the violation is on analysts' time, go to 200 (statement 24)
- 18 if the violation is on van space or cost, go to 300
(statement 25), otherwise, the violation was on equipment
time
- 19-23 searches to determine if the current method used the
equipment item on which the violation occurred and if it
does, go to 300 (statement 25). If it doesn't go to 50
(statement 9).
- 24 if the current method does not use the analyst for which
there was a violation, go to 50 (statement 9).
- 25 call subroutine SUBT
- 26-28 determines if the current method is a feasible method and
if it is, go to 500 (statement 29)
- 29-31 if the method is the last (third) feasible method, go to
550 (statement 32), otherwise, set MC to zero and increment
the level number
- 32 increment MC

33 DO loop to search all remaining levels
34 sets NPLAJ to the number of points in level J
35 DO loop to look at all feasible methods
36-37 if there is not a feasible method for the current level go
to 690 (statement 52)
38-51 call subroutines to see if new method relieves violation,
and if it does call ADD, set the violation code to zero,
and use RETURN 2. Otherwise, check another method.
52-53 set MC to one and check another level
54-55 if violation was not removed, add in time for previously
allocated method, and check another parameter
56-57 RETURN and END control statements

SUBROUTINE SUBT - This subroutine subtracts the current allocations
of equipment time, van space, analysts' times, and
cost from the previous totals

Variable Definition

same as SUBROUTINE ADD

Statements

same as SUBROUTINE ADD, except replace add with "subtract",
to with "from", increment with "decrement"

SUBROUTINE INFORM - This subroutine prints out the name of every branch and the names of all the parameters measured at each branch.

Variable Definition

AMAR (MPARM, MP2)	- array containing selected methods. AMAR (1, 1) contains the method for parameter 1. AMAR (1, 2) contains the level number at which the method is used, and AMAR (1, 3-25) contains the number of the method at the corresponding point in the level if the concentration is greater than zero
IBN (51, MBRP1)	- array containing the branch numbers for each level. IBN (1, 1) contains the number of points (branches) in level 1. IBN (1, 2-50) contains the corresponding branch numbers for each point in the level.
NPLA (25)	- contains the number of points (branches) in each level
POLN (MPARM, 5)	- contains the name of each parameter
FP (400, 4)	- contains flagged point information. FP (1,1) contains the parameter number for flagged point 1. FP (1, 2) contains the level number for flagged point 1, FP (1, 3) contains the point number for the level indicated, and FP (1, 4) contains the method used.
IFPT	- the number of flagged points
NBRNCH (MBRNC, 2)	- contains the name of each branch
BRN (MBRNC)	- contains the number of parameters measured at each branch

NP - number of parameters

NB - number of branches

Statements

1- 6 subroutine definition, COMMON and DIMENSION area

7-10 zeros out the array BRANCH and BRN

11 prints a label

12 DO loop to look at all parameters

13-14 sets NPTS to the number of points in the level obtained from AMAR

15 DO loop to look at all points

16 if a parameter is not measured at a point, go to 100 (statement 21)

17-18 sets N to the branch number and increments the number of parameters
 measured at that branch

19-21 places the parameter number in BRANCH and closes the DO loop

22 DO loop to look at all branches

23 initializes KOUNT (a flag used to determine if any measurements
 were made at a branch)

24 DO loop to look at all parameters

25 prints out the branch name the first time through the loop

26 if no more parameters were allocated at a branch, go check
 the flagged points

27 set KOUNT to indicate a measurement was made at a branch

28-30 prints the name of the parameter measured and continues the
 DO loop

31 if there are no flagged points, go to 211 (statement 40)

32 DO loop to look at all flagged points

33-35 obtains parameter number (NSUB), level number (MM), and point
 number (NN)

- 36-39 obtains the branch number, and if it is the same as the branch under consideration, print the name of the parameter, and set the flag (KOUNT) to indicate that a measurement was made at that branch, then look at the next flagged point
- 40-41 if no measurements were made at a branch, print this information and look at the next branch
- 42-50 CONTINUE statements, FORMAT statements, and RETURN, and END control statements

SUBROUTINE SAMPLE - This subroutine allocates analyst time (classification 1) for taking samples and flow measurements. The method used for sampling at each branch is determined by the user, and allocation of time is based on the highest number of samples required at a branch. Flow measurement is also determined by the user.

Variable Definition

SAMFRE (MPARM)	- sample frequency for each parameter
SUMM(6)	- contains allocations of van space, analysts' times and cost. SUMM (2) contains time allocation for analyst classification 1.
IFPT	- the number of flagged points
FP (400, 4)	- contains flagged point information. FP (1, 1) contains the parameter number for flagged point 1. FP (1, 2) contains the level number, FP (1, 3) contains the point number, and FP (1, 4) contains the method number.
IBN (51, MVRP1)	- array containing the branch numbers for each level. IBN (1, 1) contains the number of points (branches) in level 1. IBN (1, 2-50) contains the corresponding branch numbers for each point in the level.
NBRNCH (MBRNC,2)	- contains the name for each branch
NP	- number of parameters
NB	- number of branches
LENGTH	- length of the survey

BRANCH (MBRNC, MPARM)	- contains the number of the parameters measured at each branch
NSAMPL (10,5)	- contains the names of the sampling and flow measurement methods (up to 20 characters)
SAMPTI (10)	- contains the sampling and flow measurement times for the methods
SETUP (10)	-contains the set up times for the methods
NROUT (MBRNC)	- contains the code numbers of the sampling methods (first method is 1, second is 2, etc.) to use at each branch
NFLOW (MBRNC)	- contains the code numbers of the flow measurement methods to use at each branch (zero means no flow measurement)

Statements

1- 7	subroutine definition, COMMON and DIMENSION area
8-10	prints titles
11-16	reads in and prints out the sampling method names, times, and set up times
17-18	reads in the method to be used at each branch
19	DO loop to look at all branches
20	sets flag (used to indicate if a measurement is made at a branch)
21	initializes maximum sample frequency to zero
22	DO loop to look at parameters
23	if no measurements were allocated at branch 1, go to 15 (statement 28) to check flagged points

24-25 selects the parameter number and if its sample frequency is greater than the previous maximum set the maximum (SMAX) to this new frequency

26-27 set the flag to indicate that a measurement was made at the branch, and CONTINUE

28 if there are not flagged points go to 18 (statement 41)

29 DO loop to look at all flagged points

30-32 obtains parameter number (NSUB), level number (MM), and point number (NN)

33-34 obtains the branch number and if it is not the same as the branch currently under consideration, go to 16 (statement 37)

35-37 set the flag to indicate a measurement was made and determine if the new sample frequency is greater than the previous maximum. CONTINUE

38 if no measurements were made at a branch, go to 17 (statement 42)

39-41 obtain the sample method used for the branch, calculates the time required, and adds it to the previous total

42-45 print out the appropriate information

46 CONTINUE (look at another branch)

47-48 prints a title

49-51 reads in and prints out the flow measurement method names, times, and set up times

52 reads in the flow measurement code for each branch

53 DO loop to look at all branches

54 if no flow measurements are made at a branch, go to 52 (statement 61)

55-57 obtain the proper location in the •SETUP array (4 is
 added because there are 4 sample methods), calculate
 the time, and add it to the total

58-60 print out flow measurement information and look at the
 next branch

61-63 print out flow measurement information and look at the
 next branch (no flow measurements)

64-65 RETURN and END control statements

BLOCK DATA - This segment is needed to initialize variables with
 DATA statements that appear in COMMON.

SUBROUTINE PRPAR - This subroutine outputs parameter measurement information listing methods used, measure locations, expected values, and equipment and analysts' times.

Variable Definition

IFPT	- the total number of flagged points
IFPP	- the number of the next flagged point to check
FP (400, 4)	- contains flagged point information. FP (1, 1) contains the parameter number for flagged point 1. FP (1, 2) contains the level number, FP (1, 3) contains the point number, FP (1, 4) contains the method number
POLN (MPARM, 5)	- contains the equipment names
I	- the current parameter number
AMAR (MPARM, MPL)	- array containing selected methods. AMAR (1, 1) contains the method number for parameter 1. AMAR (1, 2) contains the level number at which the method is used, and AMAR (1, 3-25) contains the number of the method at the corresponding point in the level is the concentration is greater than zero.
NPLA (25)	- contains the number of points at each level
IBN (51, MBRP1)	- array containing the branch numbers for each level. IBN (1, 1) contains the number of points (branches) in level 1. IBN (1, 2-50) contains the corresponding branch numbers for each point in the level.

- PC (MPARM, 25, MBRNC) - parameter concentration at a point and a level (eg. PC (2, 4, 7) is the concentration of parameter 2 at point 7 of level 4).
- SAMFRE (MPARM) - the sample frequency of each parameter
- PMDATA (MPARM, 3, 4, 5) - array containing resource information for up to 3 methods per parameter. The information is stored as follows: first column - equipment codes (up to 4 items for each method), second column - equipment time per sample, third column - analysts' times, fifth column - cost per sample, set up time, location of analysis, and time constraint
- SMEQTI (MEQ) - sum of equipment time for each item
- EQTIME (MEQ) - total time available for each equipment item
- SUMM (6) - contains totals of van space, analysts' times and cost

Statements

- 1- 7 subroutine definition, COMMON and DIMENSION area
- 8-11 prints out titles
- 12-13 initializes LI, to indicate if there are any allocated points, and NPTS to count the total number, allocated and flagged
- 14-15 finds the allocated level at which the current parameter is measured and obtains the number of points at that level
- 16 DO loop to look at all points in the level
- 17 if a method is not used at a point, go to 150 (statement 27)
- 18-20 if it is not the first time through the loop, go to 100 (statement 25), otherwise, increment LI and NPTS

21-24 obtain the method number, print out the parameter name,
method name, measure point, and expected value, then look
at another point

25-27 print out the measure point, branch, and expected value and
look at another point

28-29 if no points were allocated, just print the parameter name

30 initialize KPP, the number of flagged points for the current
parameter that use a different method than the allocated
point (s).

31-32 if all the flagged points have been checked, or if the flagged
point is not for the current parameter, skip the next segment
by going to 200 (statement 44)

33-36 obtain the level, point, and method numbers, and the expected
concentration

37-39 increment the total number of points, and print out the
information about the current point

40 increment the flagged point array location

41-43 if the last flagged point used the same method as the
allocated point, don't increment KPP. If the method was
different, increment KPP. In either case, go check the
next flagged point.

44-46 calculate the total number of samples taken, and print it out

47-48 print out a title

49 obtain the allocated method number

50 DO loop to look at all possible equipment items for a method

51-52 if no equipment is used, go to 355 (statement 60)

53-55 calculate the time used for a method and see if the total
 time allowed has been exceeded

 56-59 print out equipment name, time, and if a violation occurred,
 then look at the next equipment item

 60 if no flagged points used a different method, go to 400
 (statement 60)

 61-62 print a label

 63 determine the method used for the flagged point

 64 DO loop to look at all possible equipment items for the method

 65-66 if no equipment is used, go to 400 (statement 75)

 67-69 calculate the time used for an equipment item by a method and
 determine if the total time allowed has been exceeded

 70-72 print out equipment name, time, and if a violation occurred,
 then look at the next equipment item

 73-76 print out labels

 77-78 obtain method numbers for allocated and flagged points

 79 DO loop to look at each analyst

 80 calculate time used for an analyst

 81-82 if there are no allocated points, go to 600 (statement 85)

 83 if an analyst is used for an allocated method calculate the
 time

 84 if the flagged and allocated methods are the same, go to 610
 (statement 86)

 85 calculate the analyst time for the flagged point

 86-88 determine if an analyst constraint was violated, and convert
 minutes to hours and minutes

 89-91 print out analyst number, time, and if a violation occurred,
 and look at the next analyst

 92-93 RETURN and END control statements

SUBROUTINE USECT - This subroutine calculates the total number of items an equipment item has been used.

Variable Definition

E (MPARM, MEQ)	- contains the code numbers of each equipment item that has been used for each parameter
NP	- number of parameters
NE	- number of equipment items
U (MEQ)	- contains the number of times each item of equipment is used

Statements

1- 2	subroutine definition and DIMENSION
3- 4	zeros out array U
5- 6	DO loop to check every equipment item and every parameter
7	if an item of equipment is used for a parameter increment the total number of times that item is used
8-10	CONTINUE, RETURN, and END control statements

SUBROUTINE TICHAN - This subroutine converts time from minutes to
hours and minutes.

Variable Definition

T - time in minutes
IHRS - time in hours
MIN - time in minutes

Statements

1 subroutine definition
2 converts from real to integer
3 converts minutes to hours (any remainder in the division
 is truncated)
4 calculates the number of minutes remaining
5-6 RETURN and END control statements

SUBROUTINE CORRCT - This subroutine accepts branch measurement data and computes process model parameters from these branch measurements. The output compares the process model parameters to computed process parameters. The subroutine is called from MAIN if model update or verification is desired.

Variable Definitions

A(MSORS, MBRNC)	- contains the topology matrix
NS	- number of sources in a particular study
NP	- number of parameters in a particular study
NB	- number of branches in a particular
C(MSORS, MPARM)	- contains the parameters concentrations for NS sources and NP parameters
NPLIST(MPARM)	- lists the numbers (from the master list) of the parameters in this study
EFF(MBRNC)	- contains the flow for each branch
X(MBRNC,MPARM)	- contains the mass parameters for each of the NB branches
FLOW(MSORS)	- contains the flow for each source in MGD
P(MSORS, MPARM)	- contains the mass parameters for each source [C(J,K)*FLOW(J)]
Y(MBRNC, MPARM)	- parameter concentrations for each of NB branches and NP parameters
POLN(MPARM,5)	- parameter names
MSORS	- program dimension for the number of sources
MPARM	- program dimension for the number of parameters
MBRNC	- program dimension for the number of branches

MESUR(25)	- lists the branches for which there are measurement data
BPLIST(25,20)	- For up to 25 branches and 19 parameters lists parameter numbers (from master list) for which measurements are taken. Except BPLIST(IP,1) which lists the number of parameters measured.
YM(25,25)	- contains measure values for flow and other parameters for up to 25 branches.
Statements	
1-7	subroutine definition and dimensions
8-9	reads out NB,NP and NS
10-11	Reads in MESUR(1)
12-19	Reads in BPLIST(IB,N), YM(IB,N)
20	Sets UN = 0 counts the uniquely determined sources
21	Sets NUN = 0 counts the non-uniquely (estimated) determined sources
22-129	Scans the branches for measurements
23	If no measurement, next branch
24	If all sources have been determined, read out summary
25	IFLAG = 0 treatment flag
26	IC = 0 counts sources contributing to branch if measurements were taken
27-39	counts and identifies sources contributing to branch if measurements were taken. If any source flow has been treated upon arriving at branch 1, IFLAG = 10
40	If only one source contributes to branch go to 11(51)

40-47	Count the contributing sources which have not been previously determined
48	If only one go to 15(84)
49	If equal to zero-next branch
50-51	If more than one-go to next branch for now
52-56	If the one contributing source has already been determined, go to next branch
57-59	If not, set MESUR(1) = 2 add one to UN and identify newly determined source UNIQ(UN)
60	Set IBPLST equal to the number of parameters measured at branch 1
61-69	Set flow of this newly determined source equal to the measured flow and read out
70-71	identify parameter measured next BPLIST(1,IP)
72	if treated go to 1210(77)
73-76	Set parameter of this newly determined source equal to the measured value for branch 1 and read out values
77-81	if treated compute parameter of newly determined source and read out
82	look at next parameter IP
83	when all parameters included in branch 1 have been considered consider next branch
84	arrive here if there is more than one source contributing to branch 1
85-91	Identify the contributing source that has not been previously determined

91-93	Add one to UN, identify newly determined source UNIQ(UN)
94	Set MEASUR(I) = 2
95	!BPLST equal to number of measurements on branch I
96-	consider each measurement (IP) on branch I
97-115	if measurement is flow, subtract known contributing source flows from measured flow in branch I and read out data. Then look at next parameter.
116-117	identify next parameter
118-127	for this parameter, subtract known contributing sources mass values from the total parameter mass in measured branch. This leaves mass contributed by new source. Convert to concentration. read out.
128	look at next parameter for branch I
129	look at next branch
130	if all sources have been determined go to final summation
131-179	look at each process J
132-134	has the Jth process been determined if so look at next process
135	set treatment flag IFLAG = 0
136-139	find first branch in which source J appears
140	if this branch was not measured look at next source
141	if branch was measured, is it treated, if so set IFLAG = 10
142-143	set counters IC1 = 0 and IC = 0 (contributing sources in branch I and branch (I - 1))
144-148	count and identify sources contributing to branch I

149-155	count and identify sources contributing to branch II = I - I
156-158	if the difference in the number of contributing sources in branch I and II is greater than 1, look at next source
159-166	count and identify sources contributing to branch I but not to II.
167	if this is greater than 1, next source
168	if this is not the source, J, under current consid- eration, look at next source
169-171	if this source is the current source, J, then set MESUR(I) = 2, UN = UN + 1 and UNIQ(UN) = J
172	are the same number of parameters measured in I and II. if not go to next source
173	if yes set IBPST = number of parameters measured
174-196	look at each parameter
175	if parameter is not flow go to 750(182)
176	if parameter is flow compute flow of newly determined source as the difference in measured flow in branches I and II.
177-180	read out flow calculation results
181	next parameter
182	if parameter is not flow continue from here
183-184	identify parameter BPLIST(IP)
185	if treated go to 790(191)
186-187	if not, calculate the mass contribution of newly determined source as the difference in mass from measurements at branch I and II. Convert to con- centration.

188-189	read out results of calculations
190	next parameter
191-194	arrive here if treated. Compute mass contribution of new source as the same % contributed by model for each parameter (IP)
195-196	read out results, look at next parameter
197	after all parameters look at next source J. after all sources have been scanned, continue.
198	if all sources have been determined go to summary read out
199	set counter of nonunique (NUN) or estimated sources
200-249	scan all branches again
201	if branch I was not measured or if measurement was used previously go to next branch
202-208	otherwise, count and identify contributing sources, IC and ICONS(IC)
209	set counter NUNPR = 0 undetermined branch sources
210-223	count and identify undetermined sources in current branch and in system
224	if no newly undetermined source go to next branch
225	IBPLST equals parameters measured
226-246	look at each parameter IP
227	if IP is not flow go to 561(235)
228-232	if flow, compute flow of each contributing undetermined source as PER times flow measured in branch I. PER is the decimal fraction contributed by that source in the model. read out results

233	next contributing source
234	next parameter after flow
235-244	for parameter k, compute mass of each contributing undetermined source as PER times mass measured in branch l. PER is the decimal fraction contributed by that source in the model. Convert to concentra- tion and read out.
245	next contributing source
246	next parameter
247	total of unique and nonunique determined sources.
248	if all sources have been determined go to final summation.
249	next branch
250-260	list the uniquely determined sources and list the non- uniquely determined sources.
261-262	return-end.

AD-A036 524

CLEMSON UNIV S C COLL OF ENGINEERING

F/G 13/2

A SYSTEMS ANALYSIS OF WATER QUALITY SURVEY DESIGN. APPENDIX III--ETC(U)

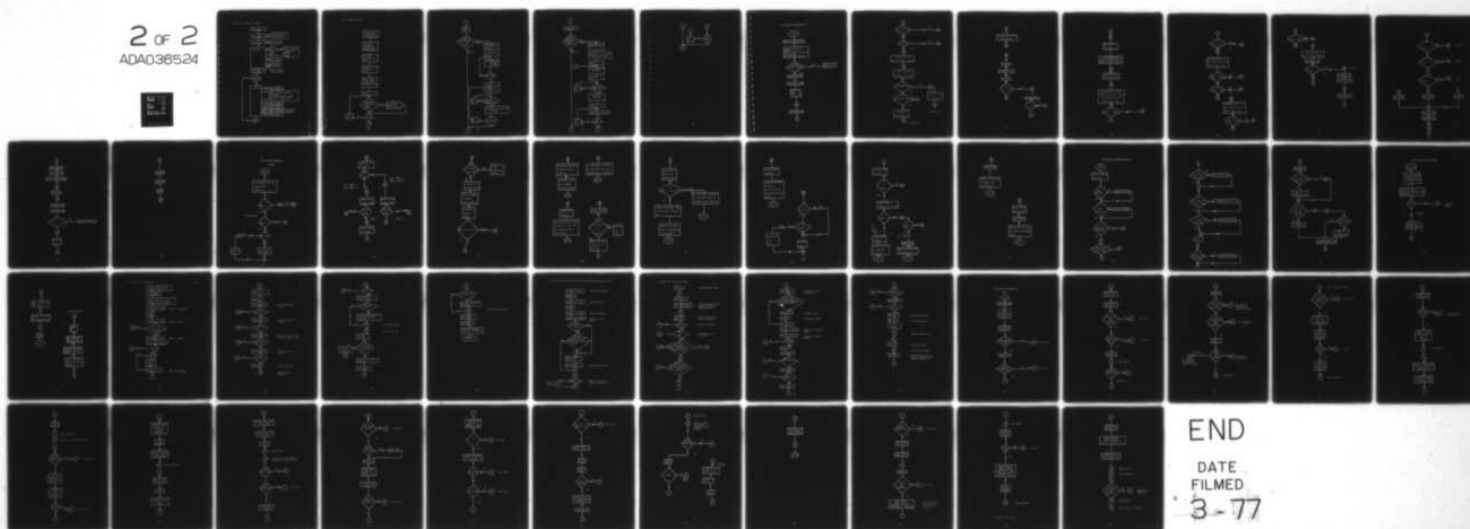
AUG 75 L C WILCOX, B E GILLILAND

DADA17-72-C-2152

NL

UNCLASSIFIED

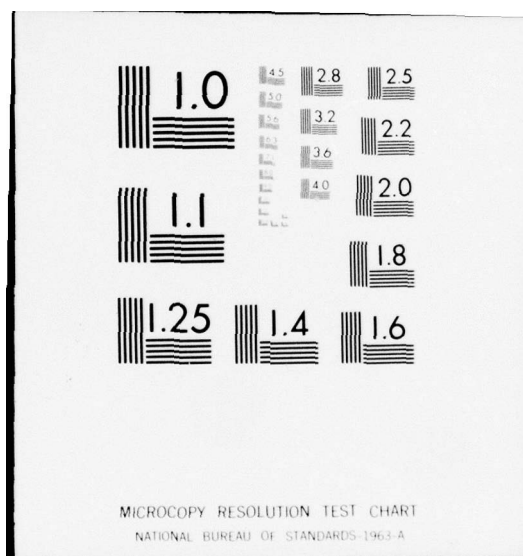
2 of 2
ADA036524



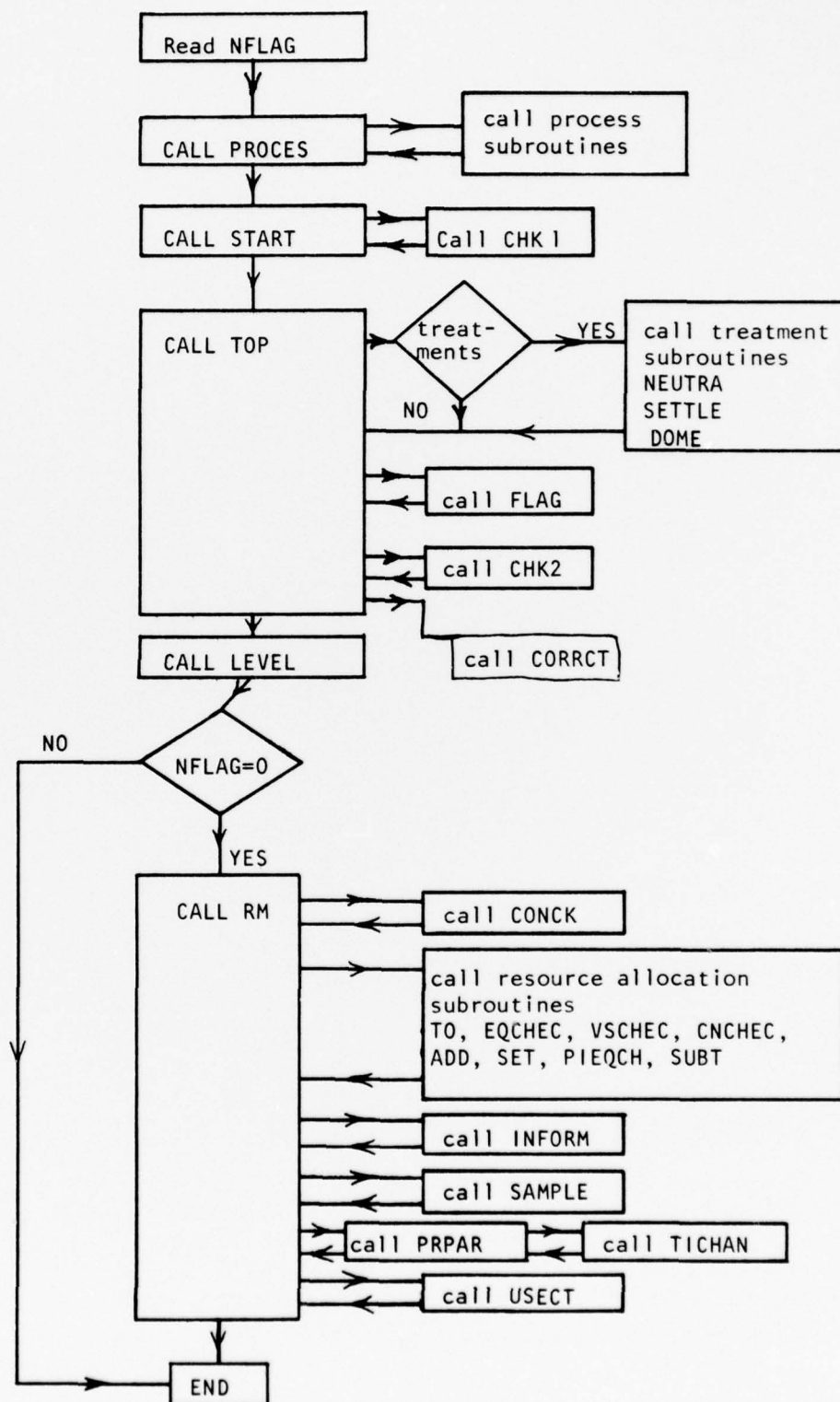
END

DATE
FILMED

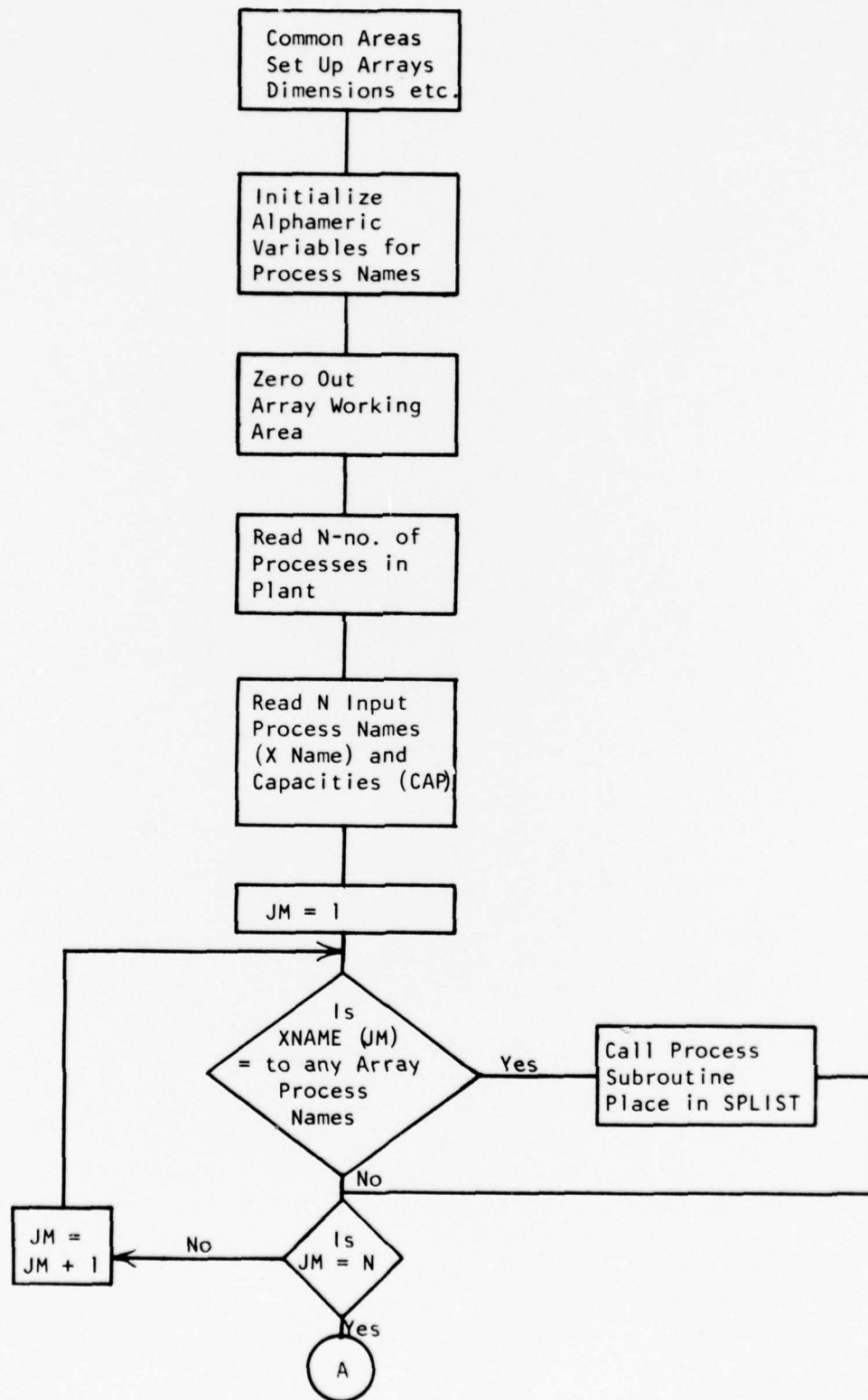
3-77

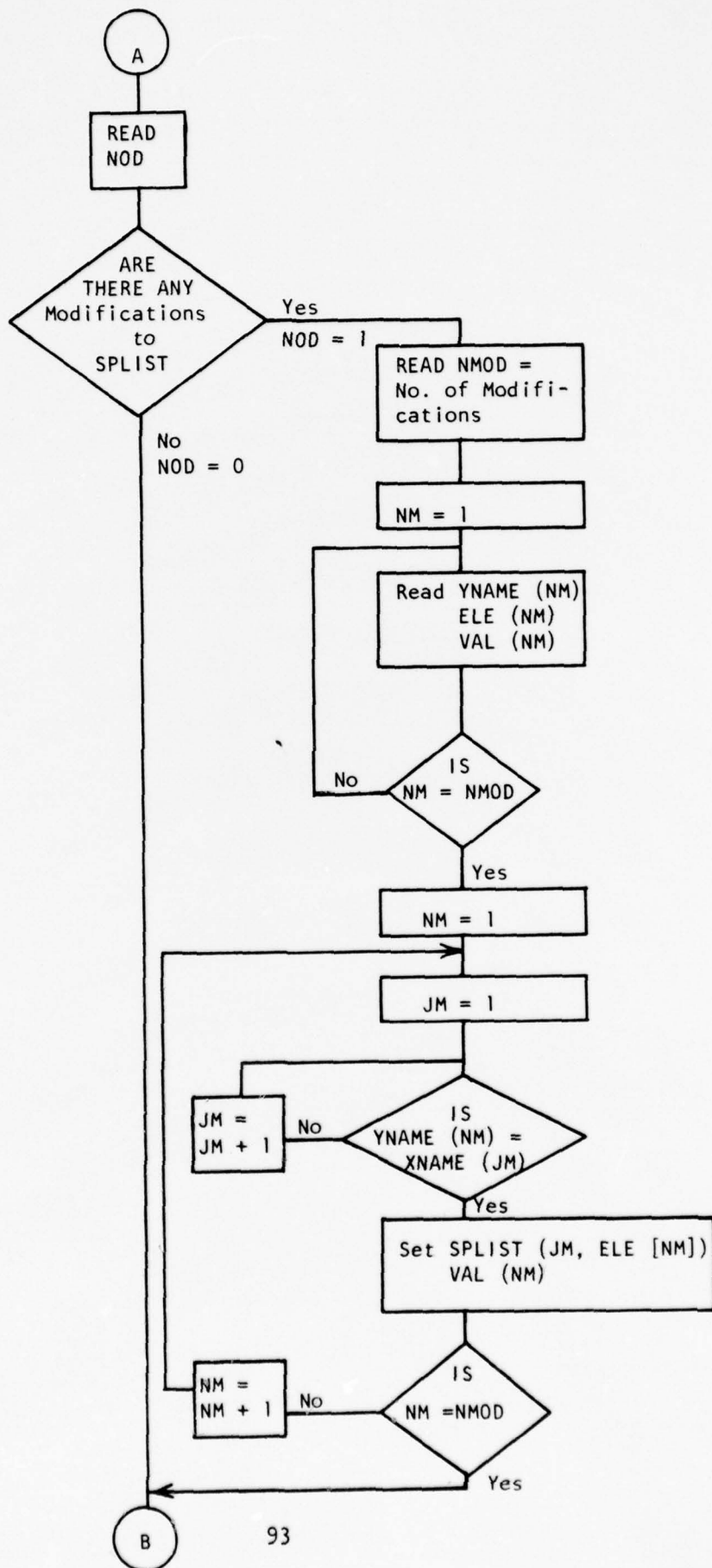


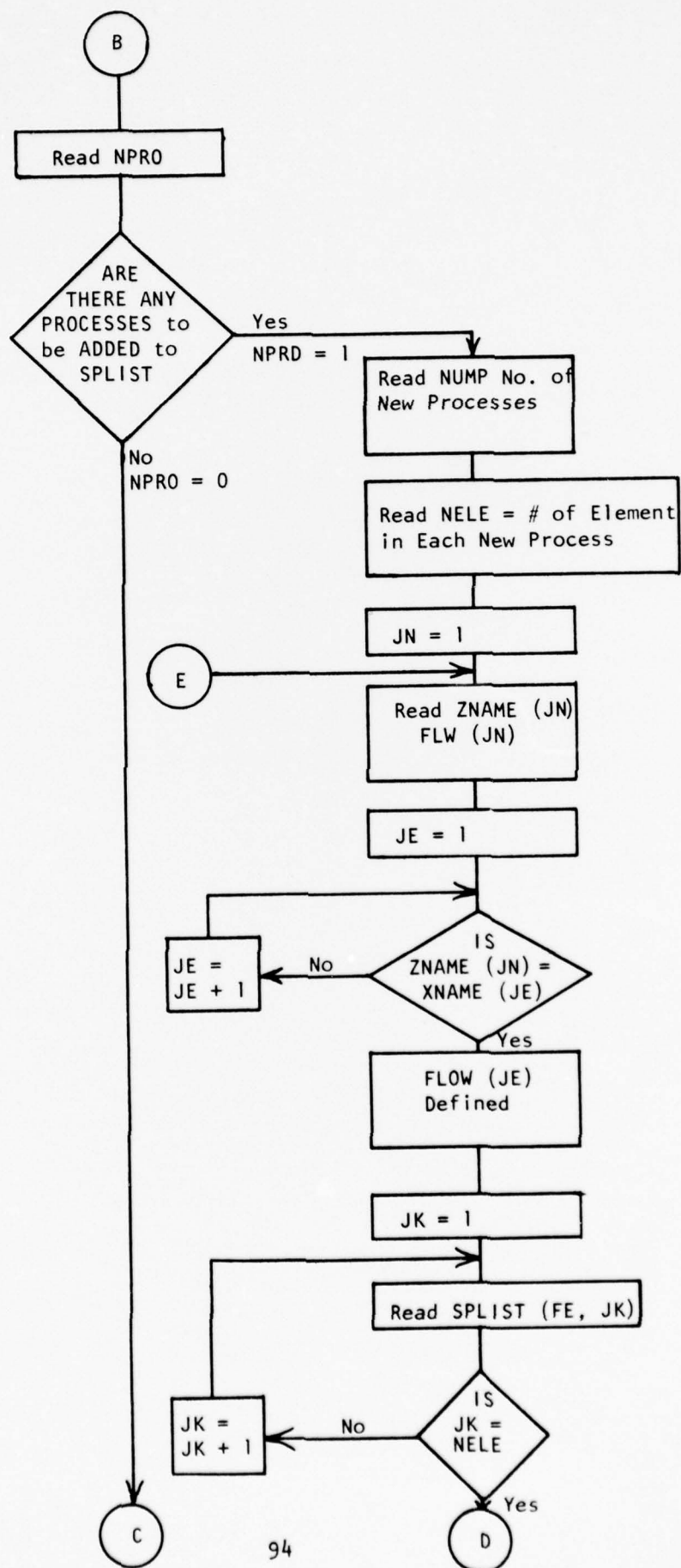
Flow Chart of Composite Program

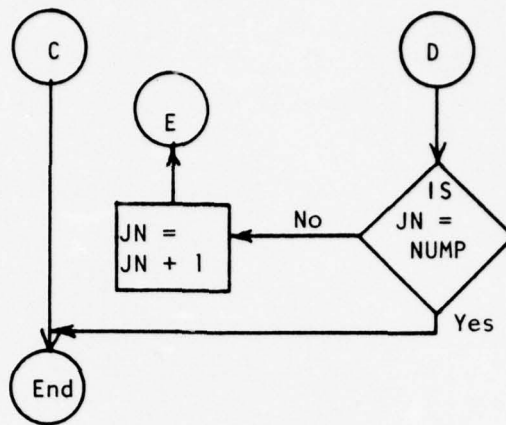


Process Model Flow Chart

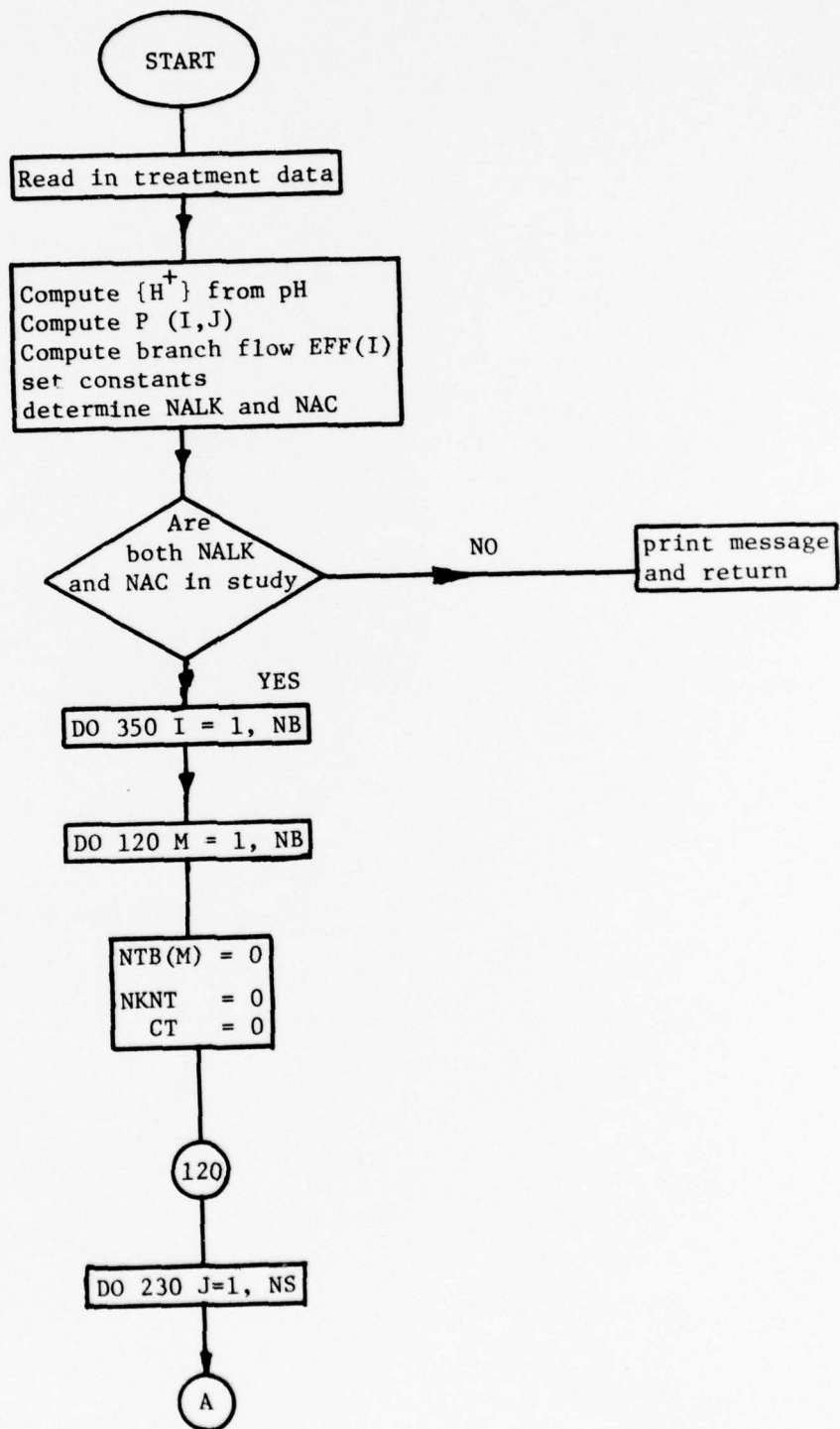


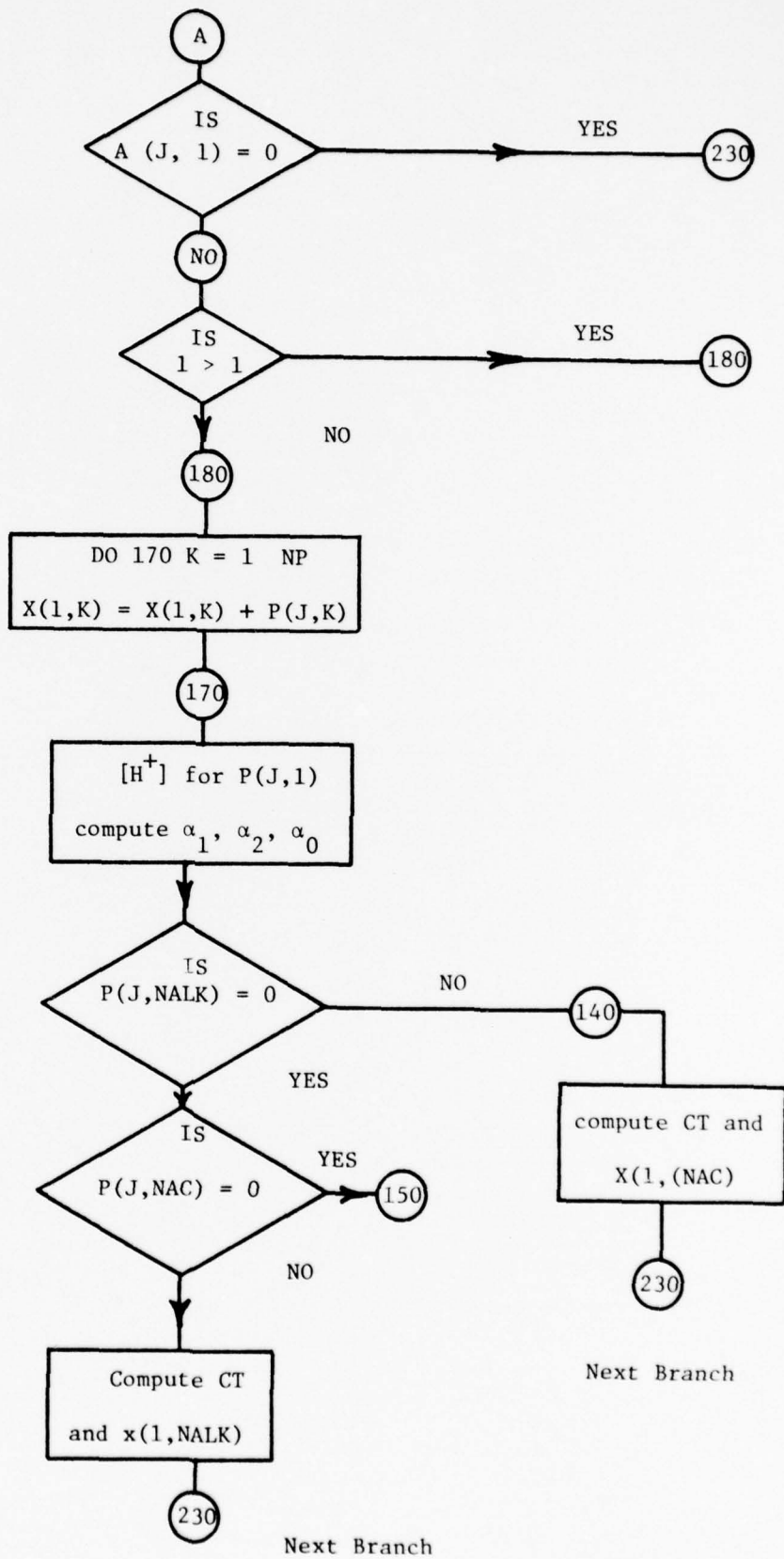


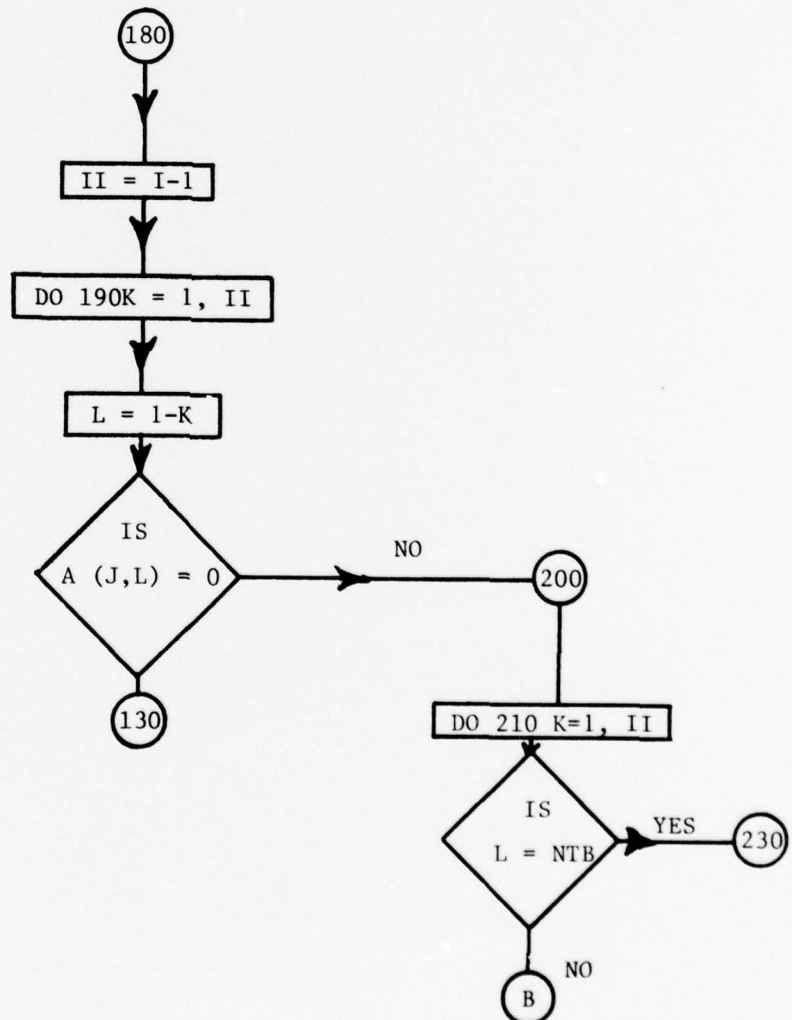
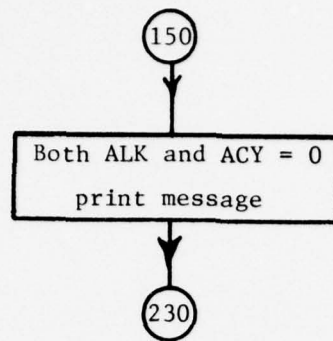


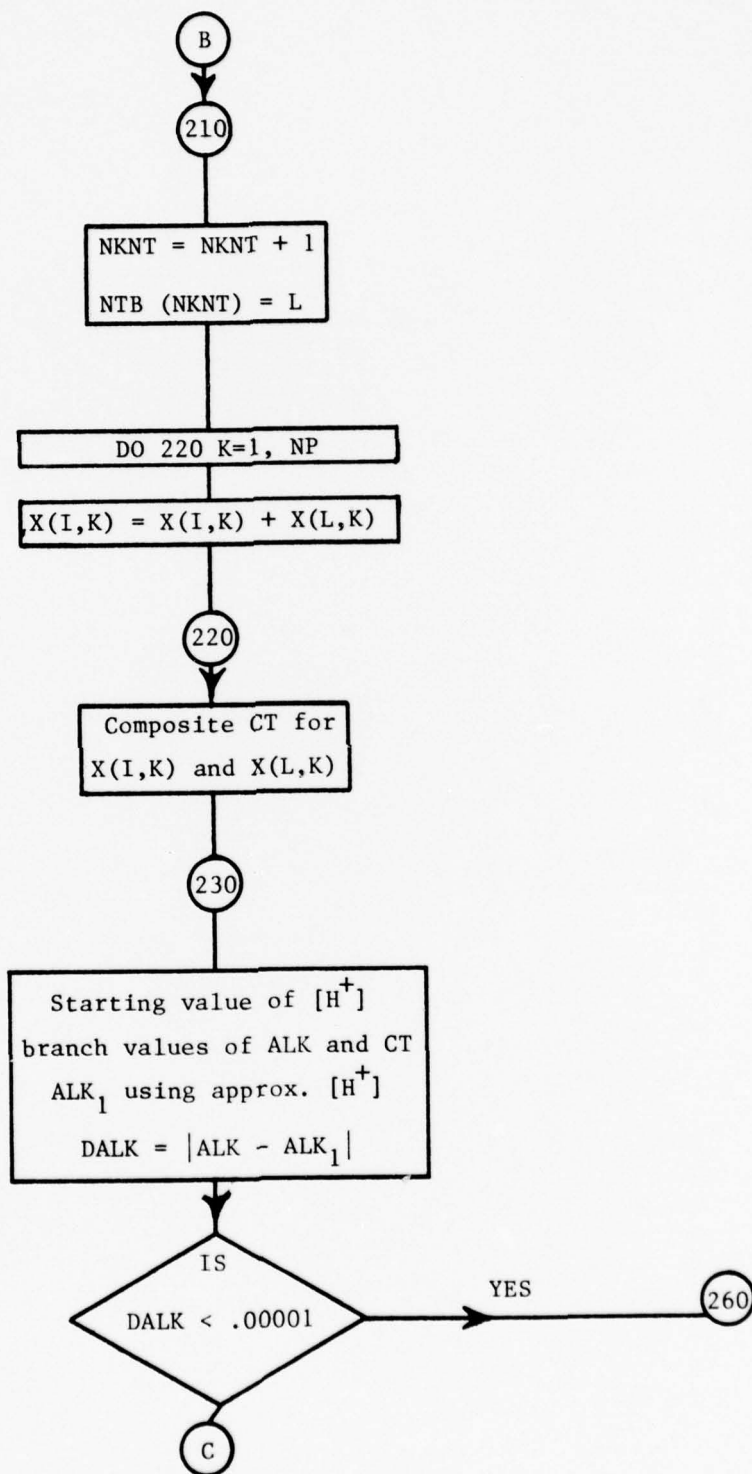


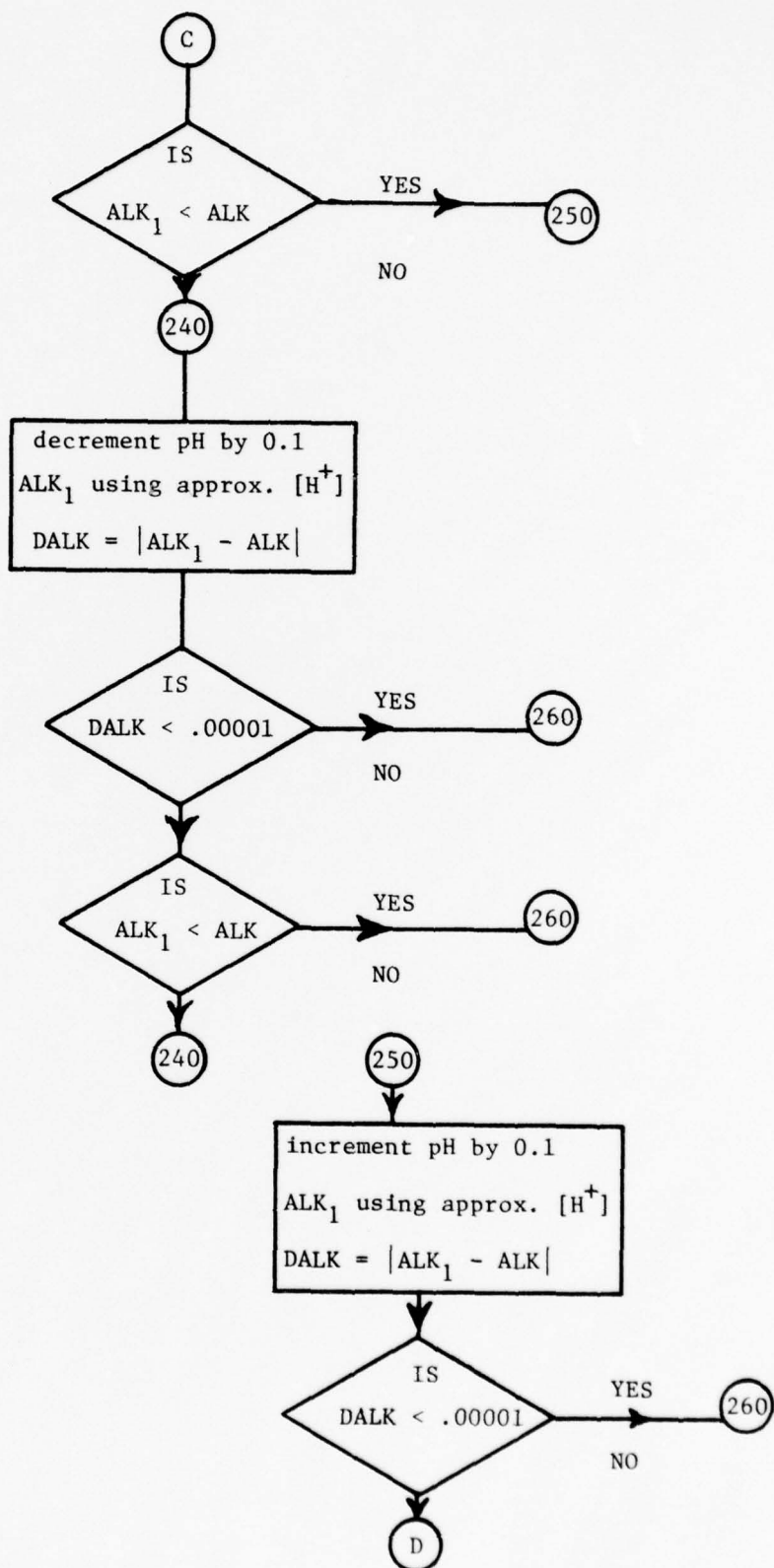
FLOW CHART FOR SUBROUTINE TOP

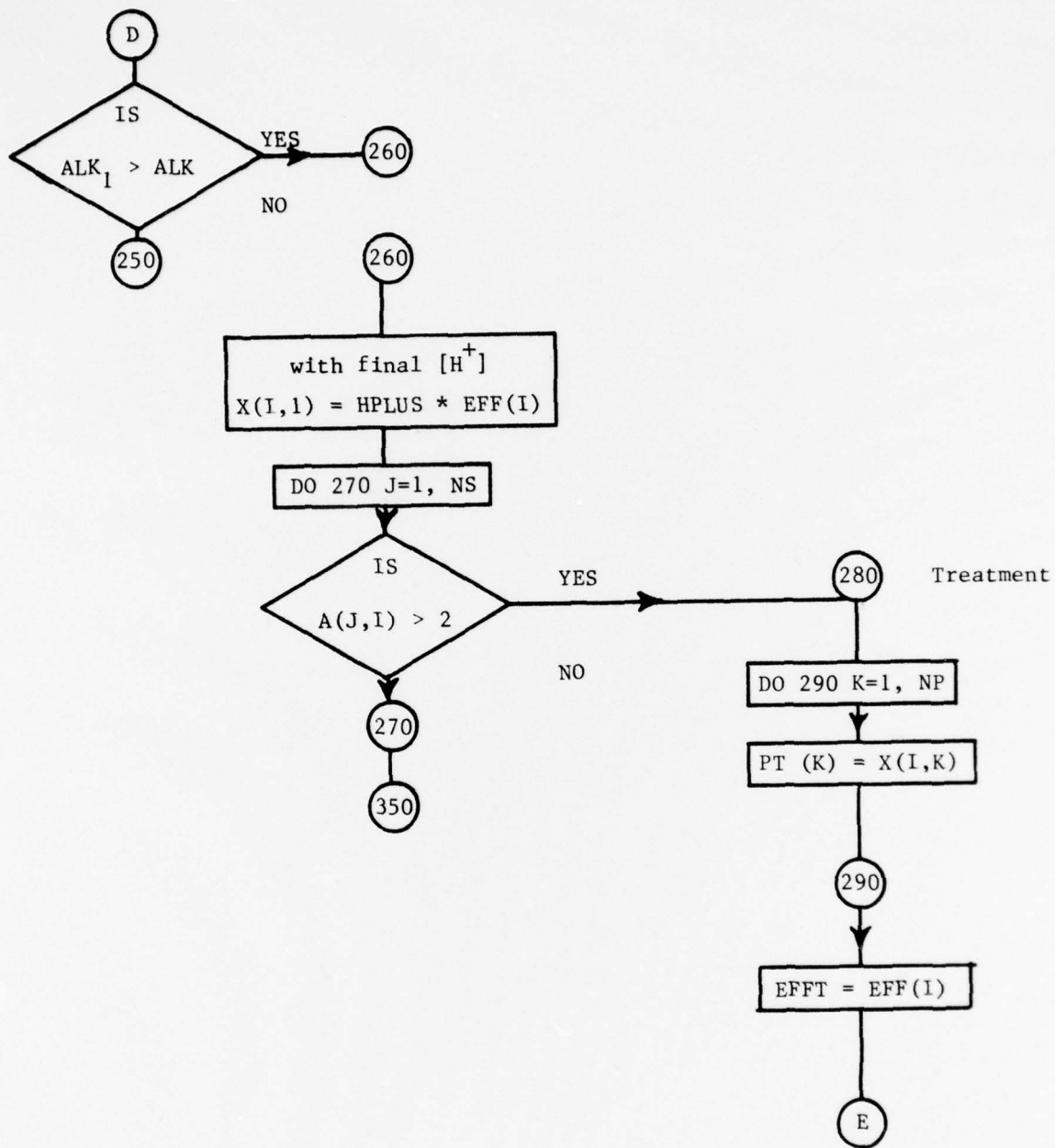


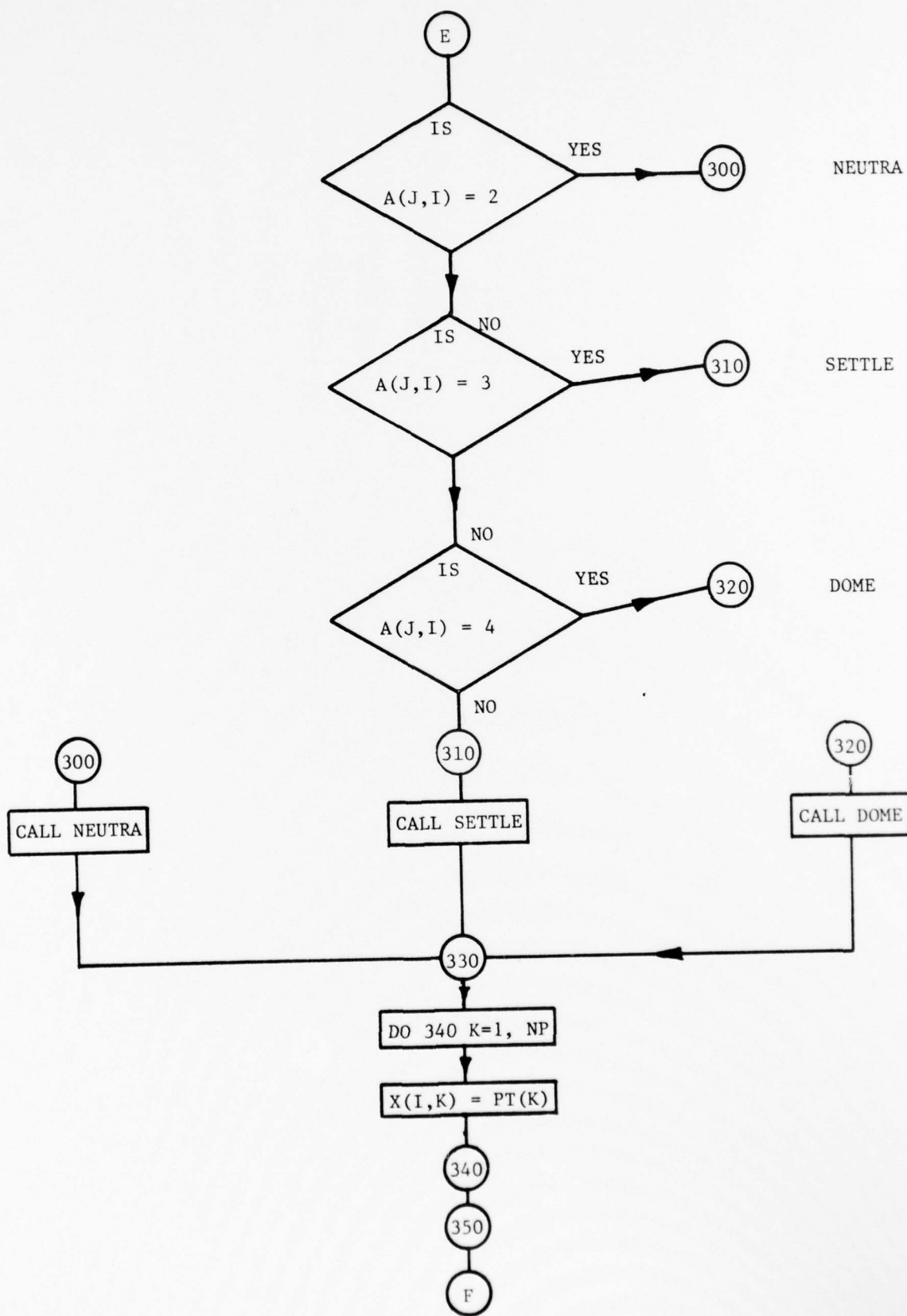


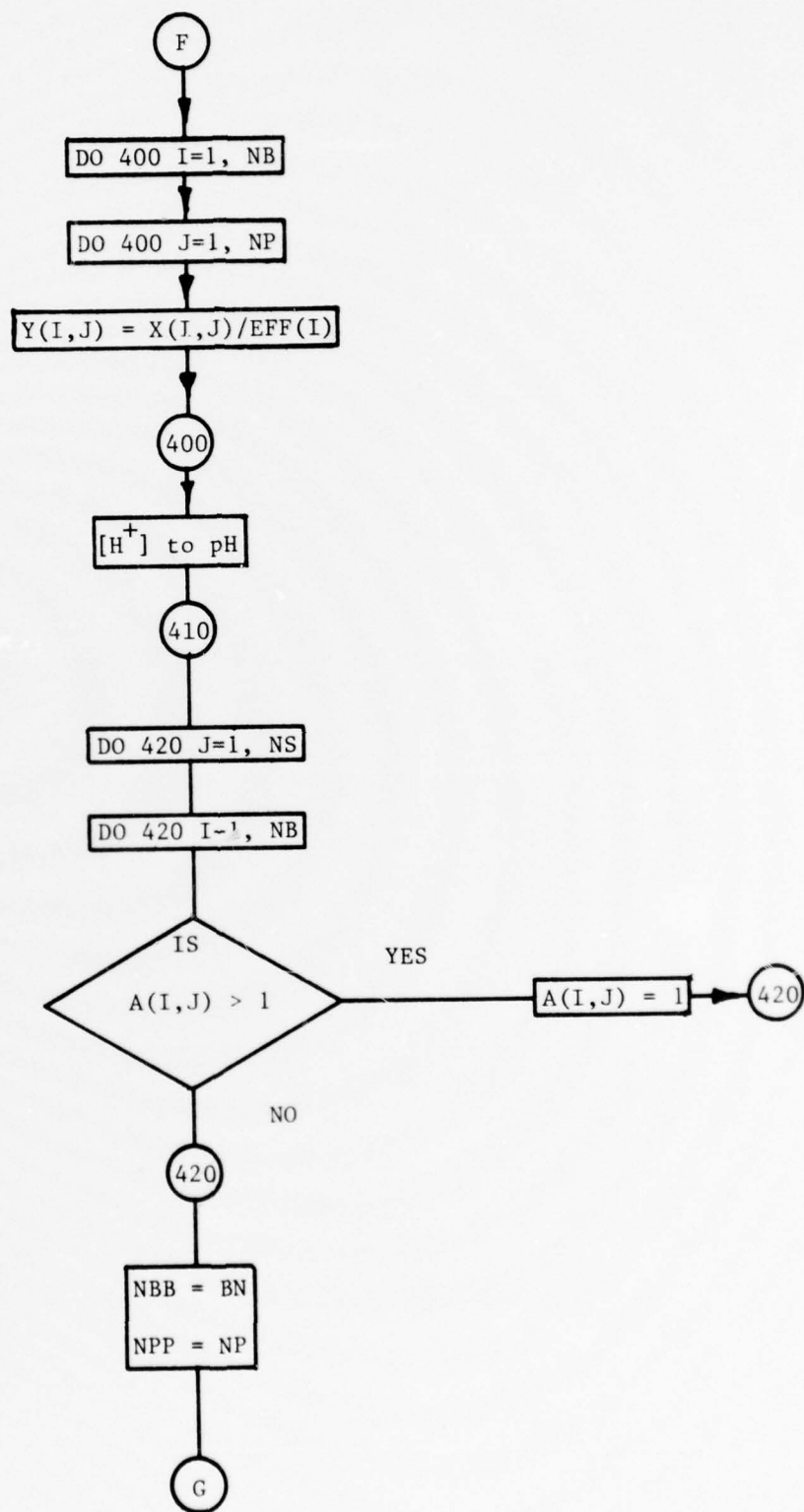








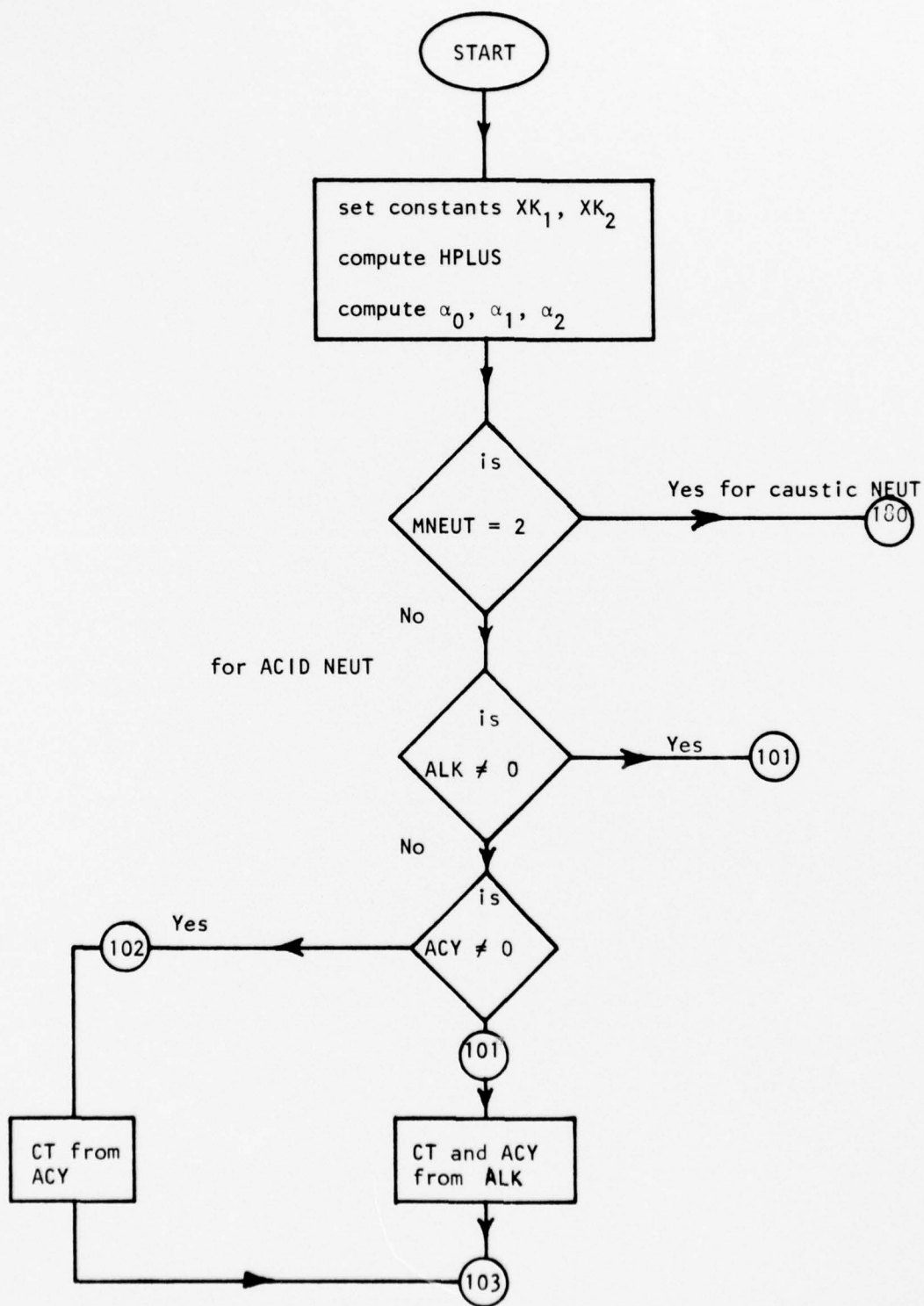


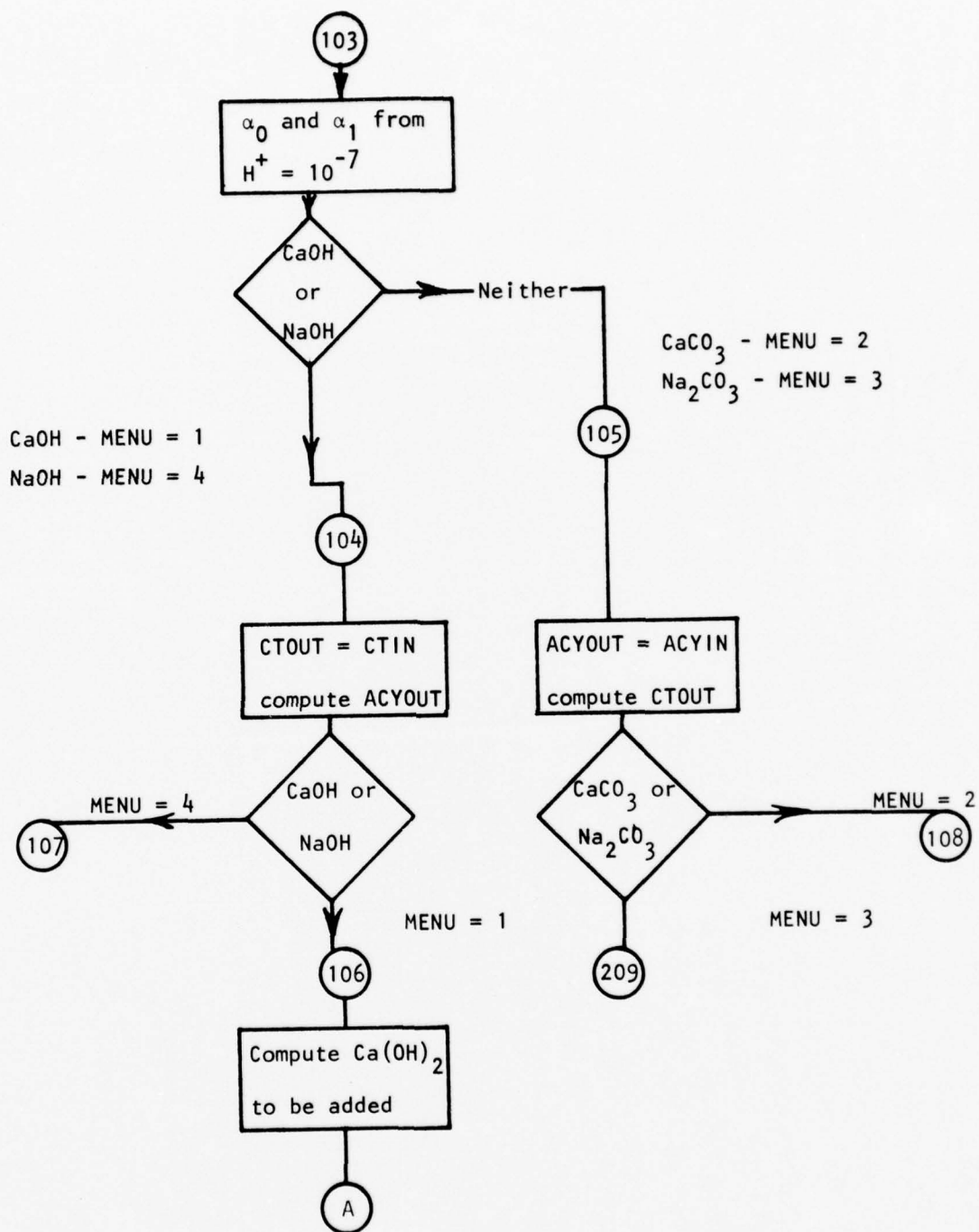


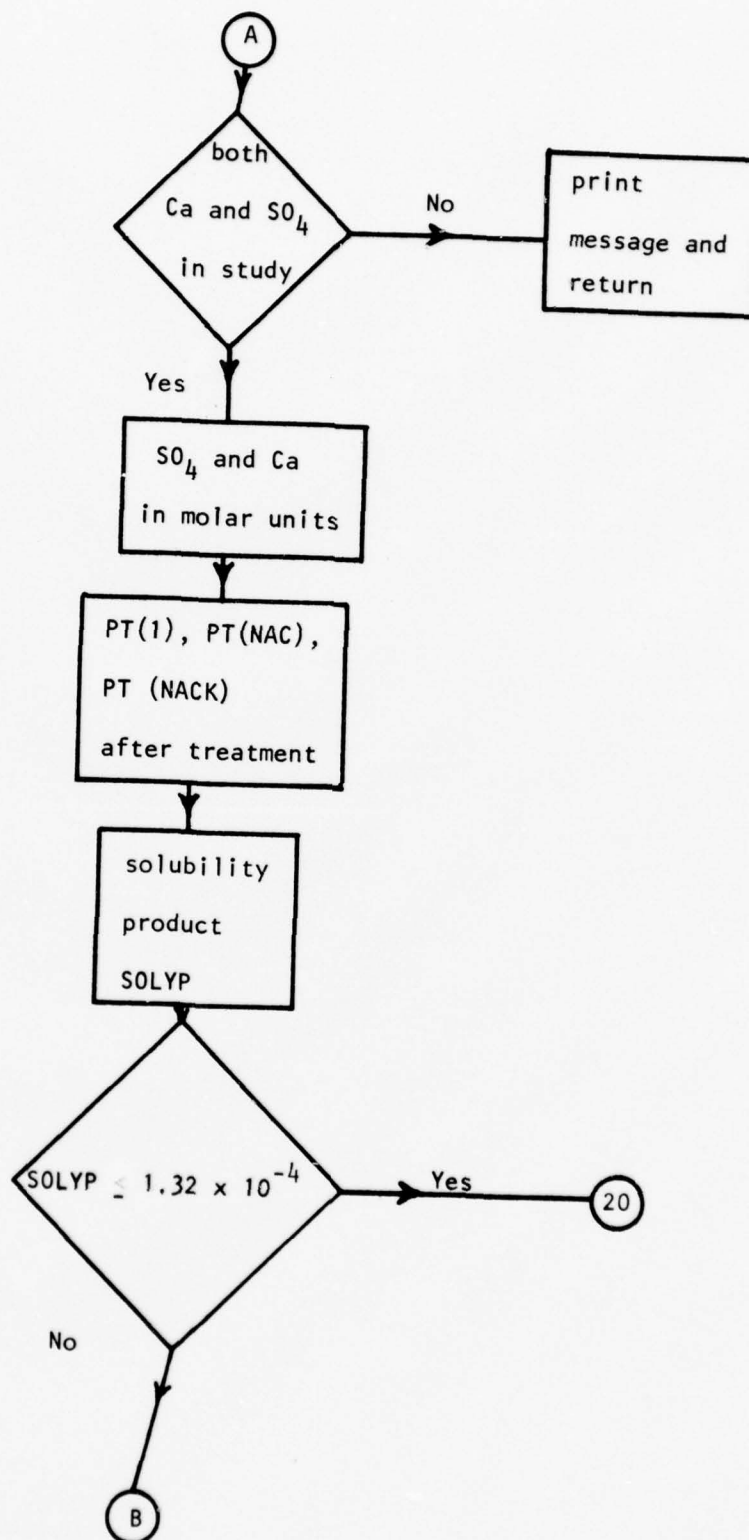


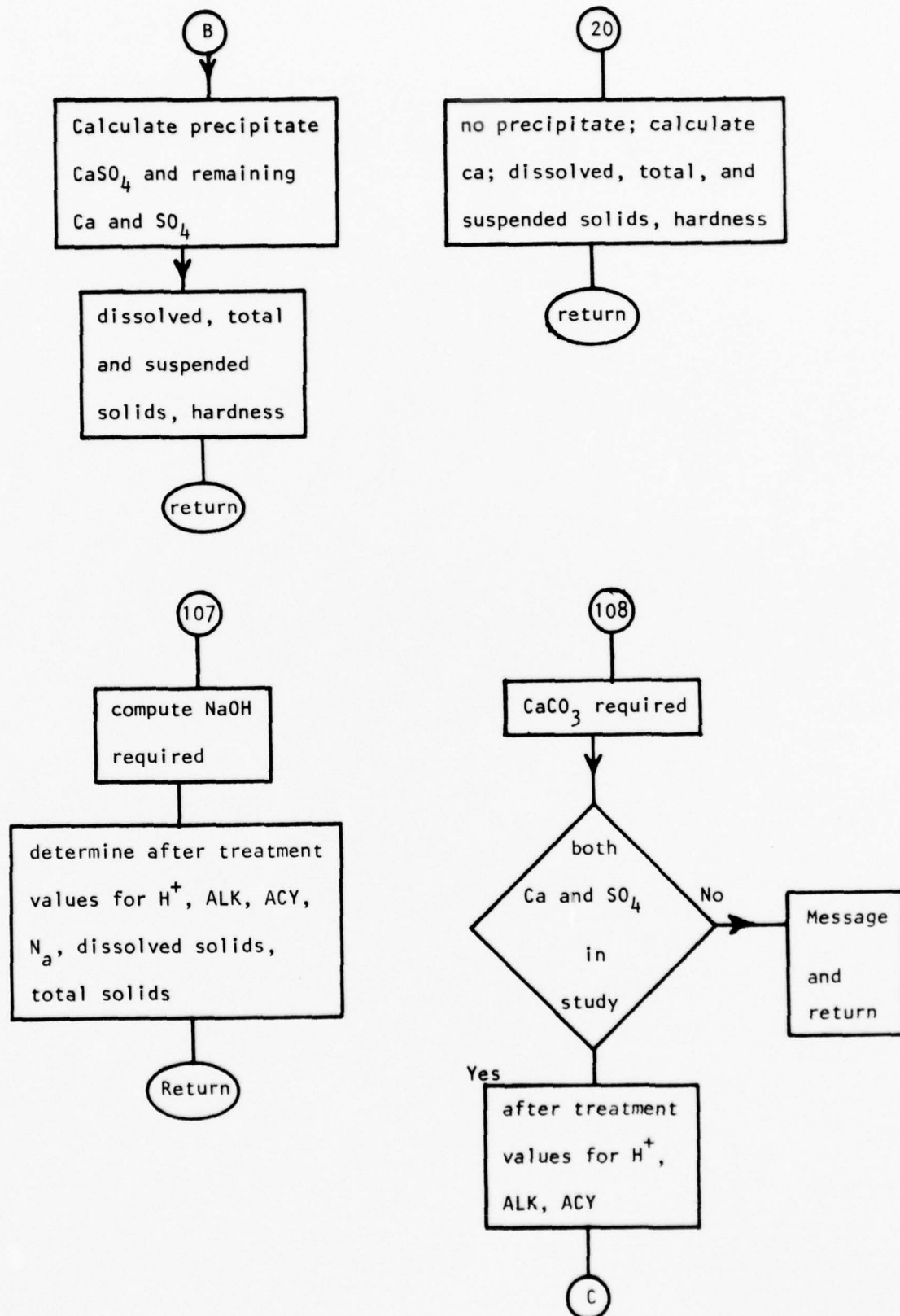
FLOW CHART FOR SUBROUTINE

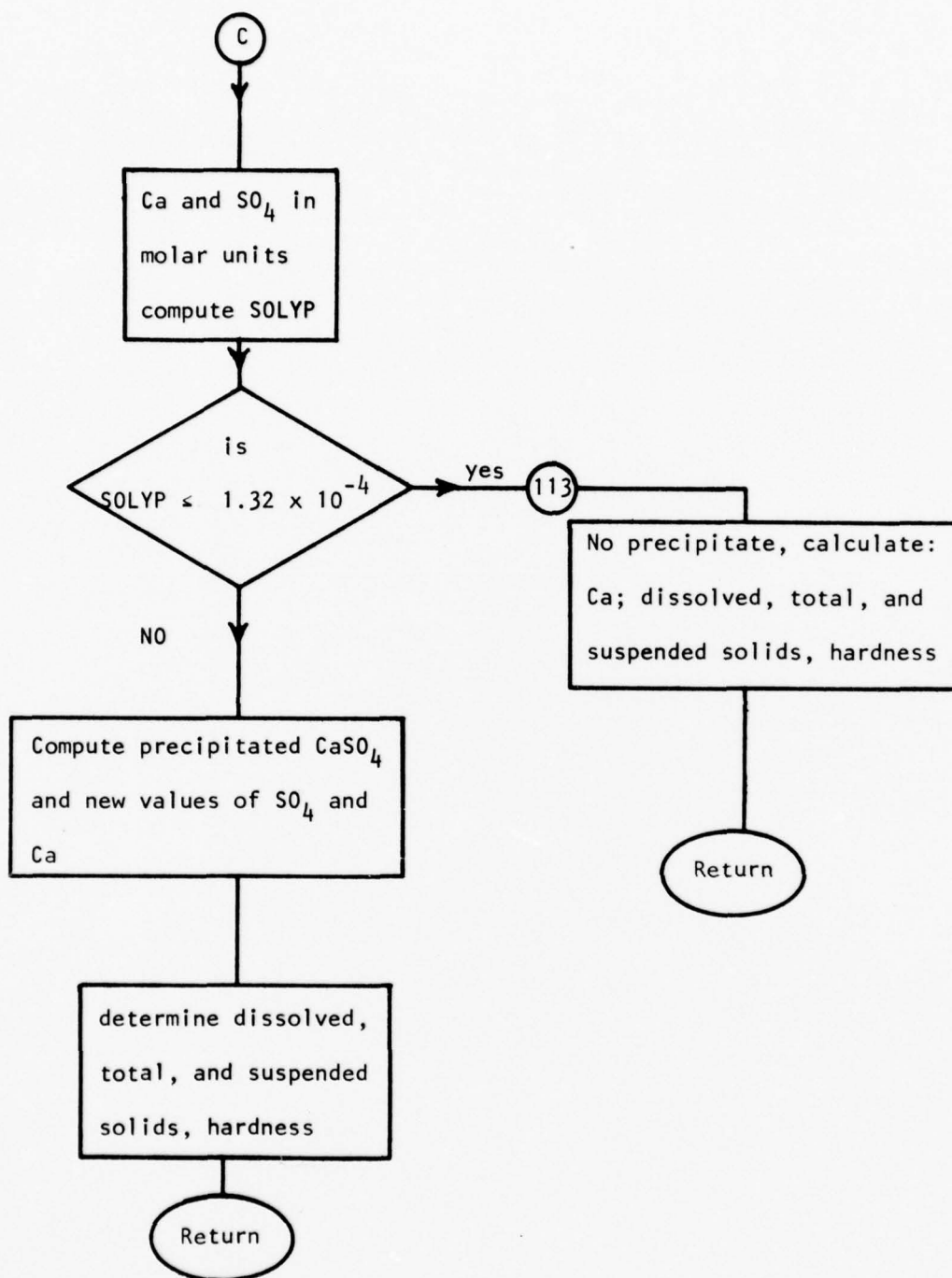
NEUTRA

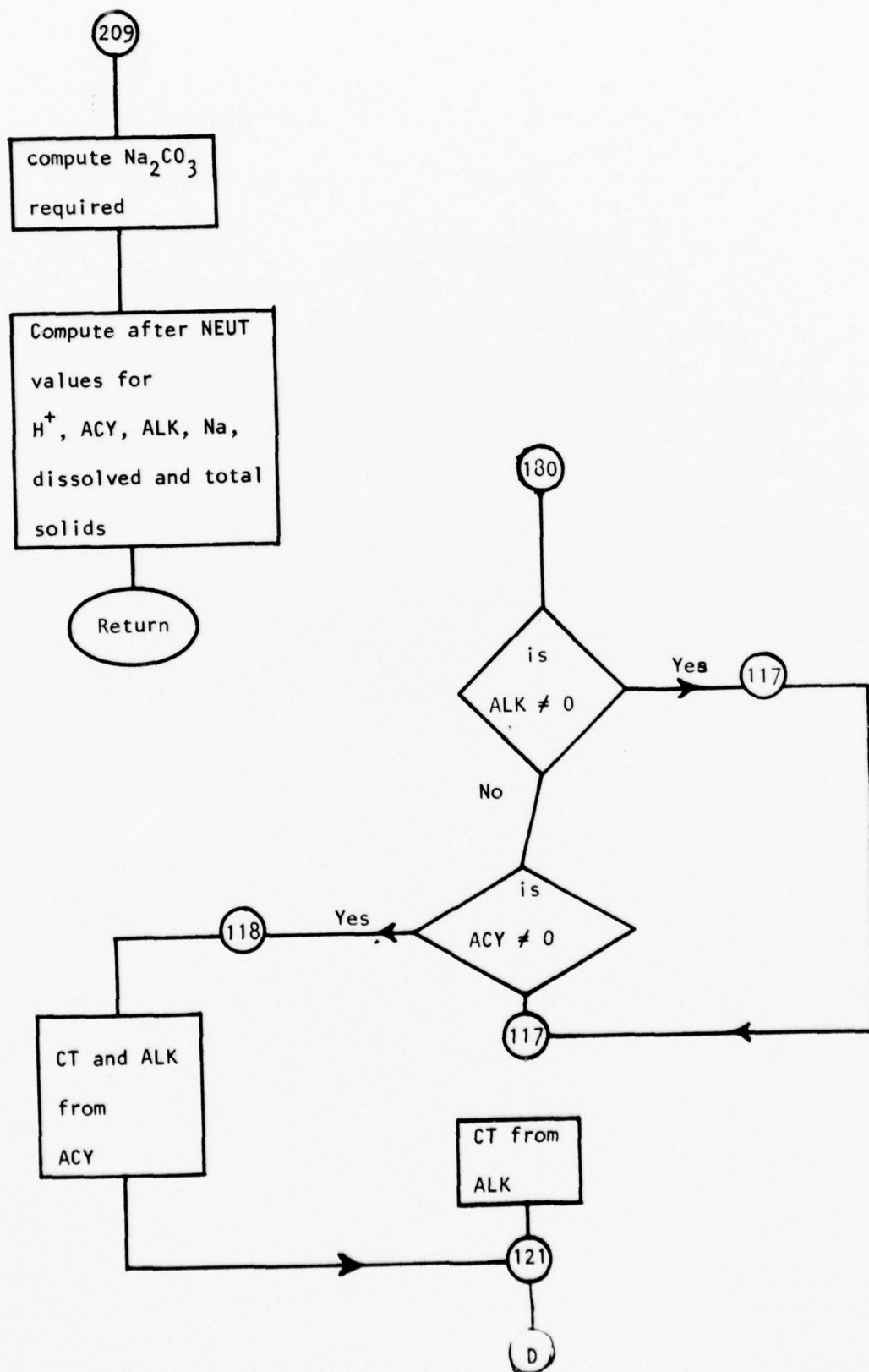


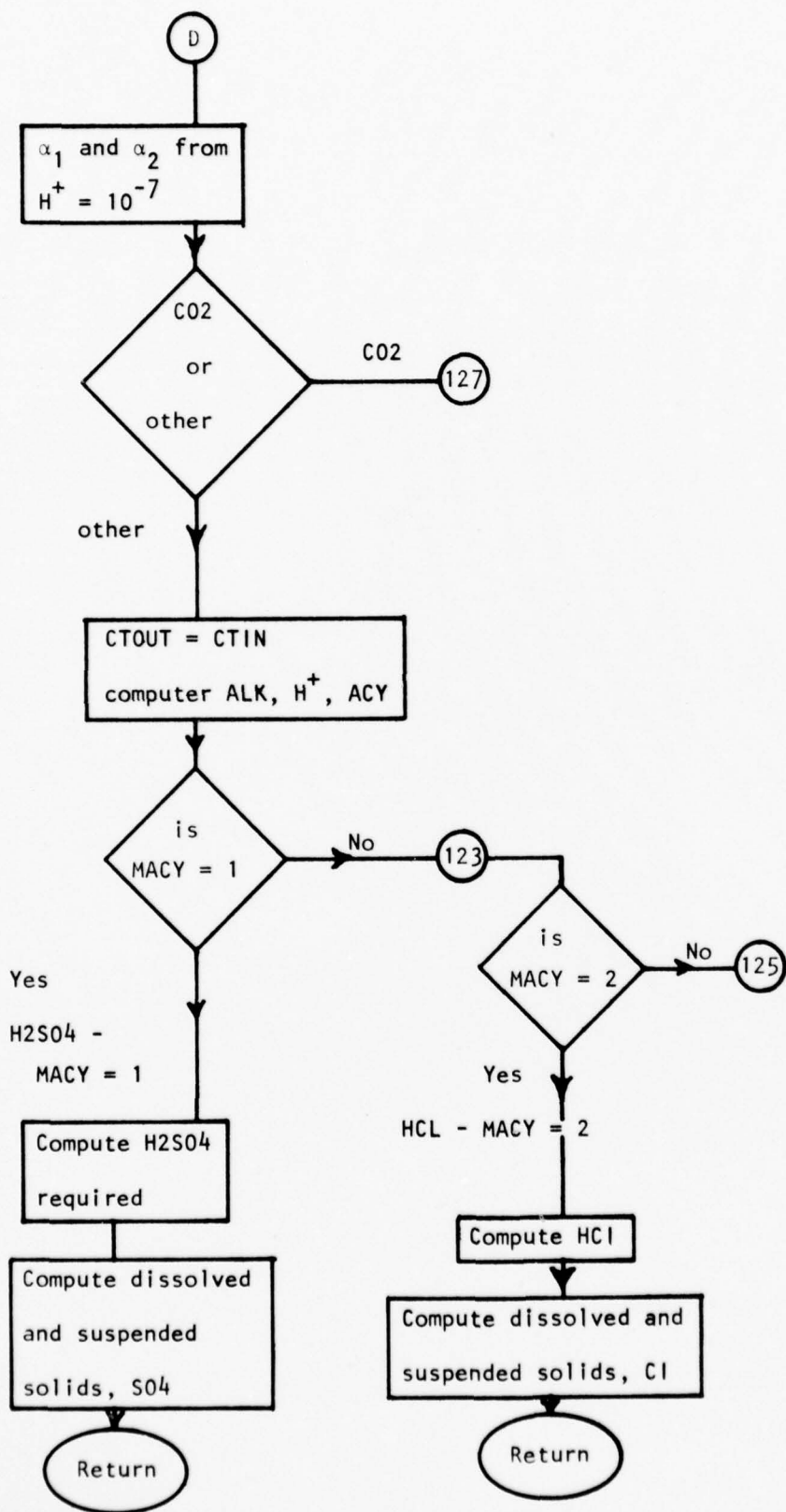


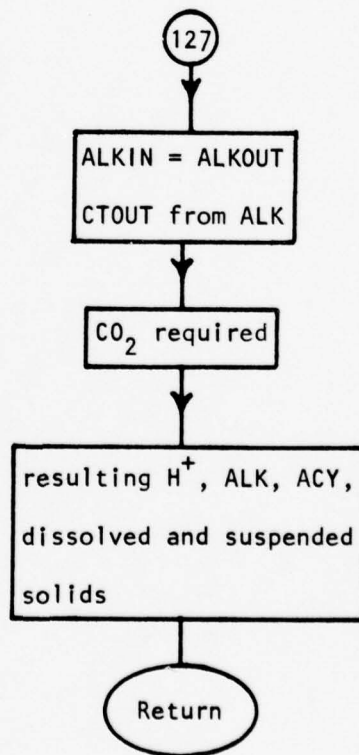
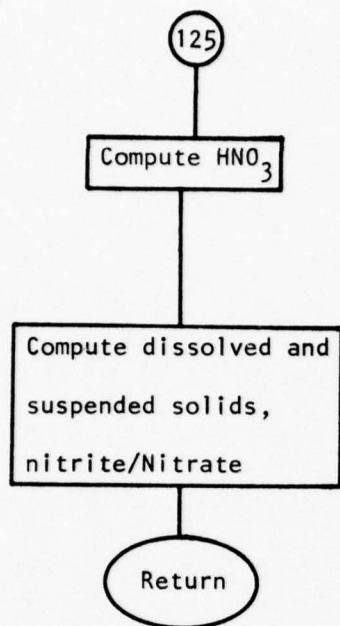




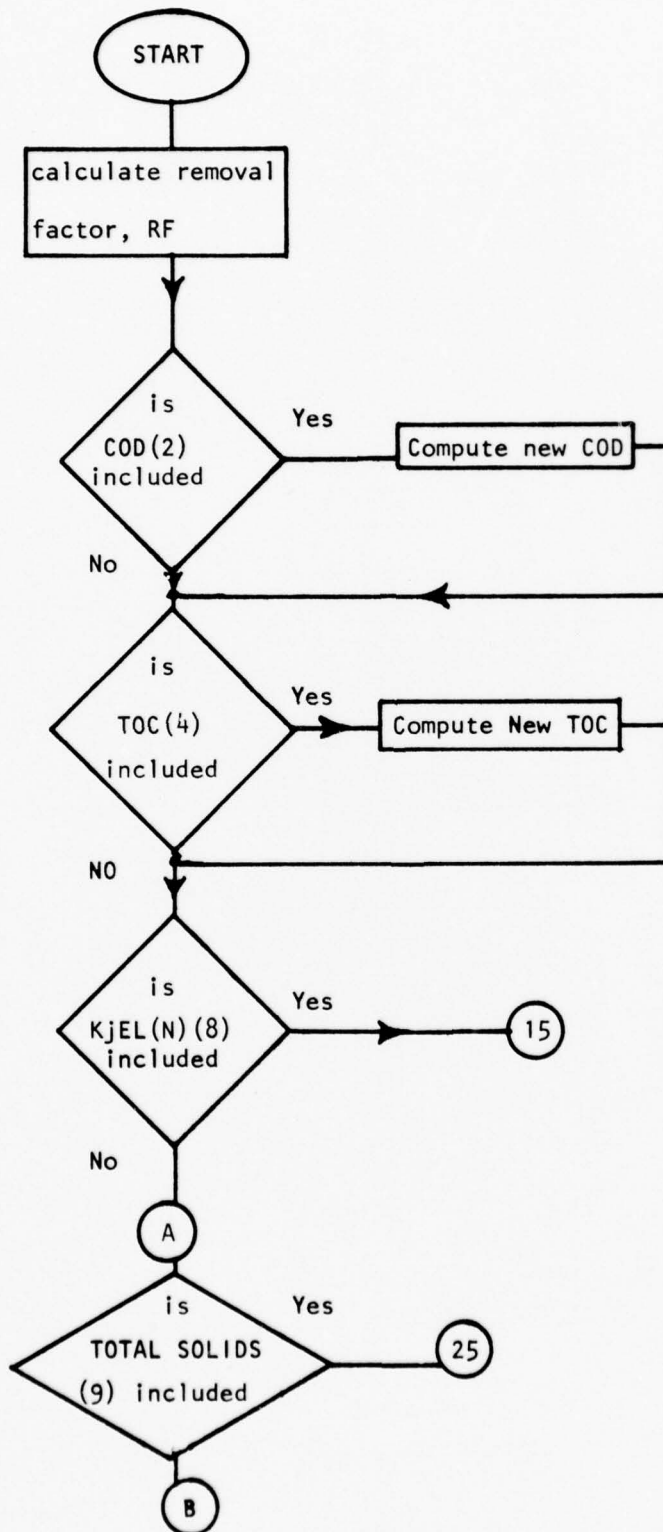


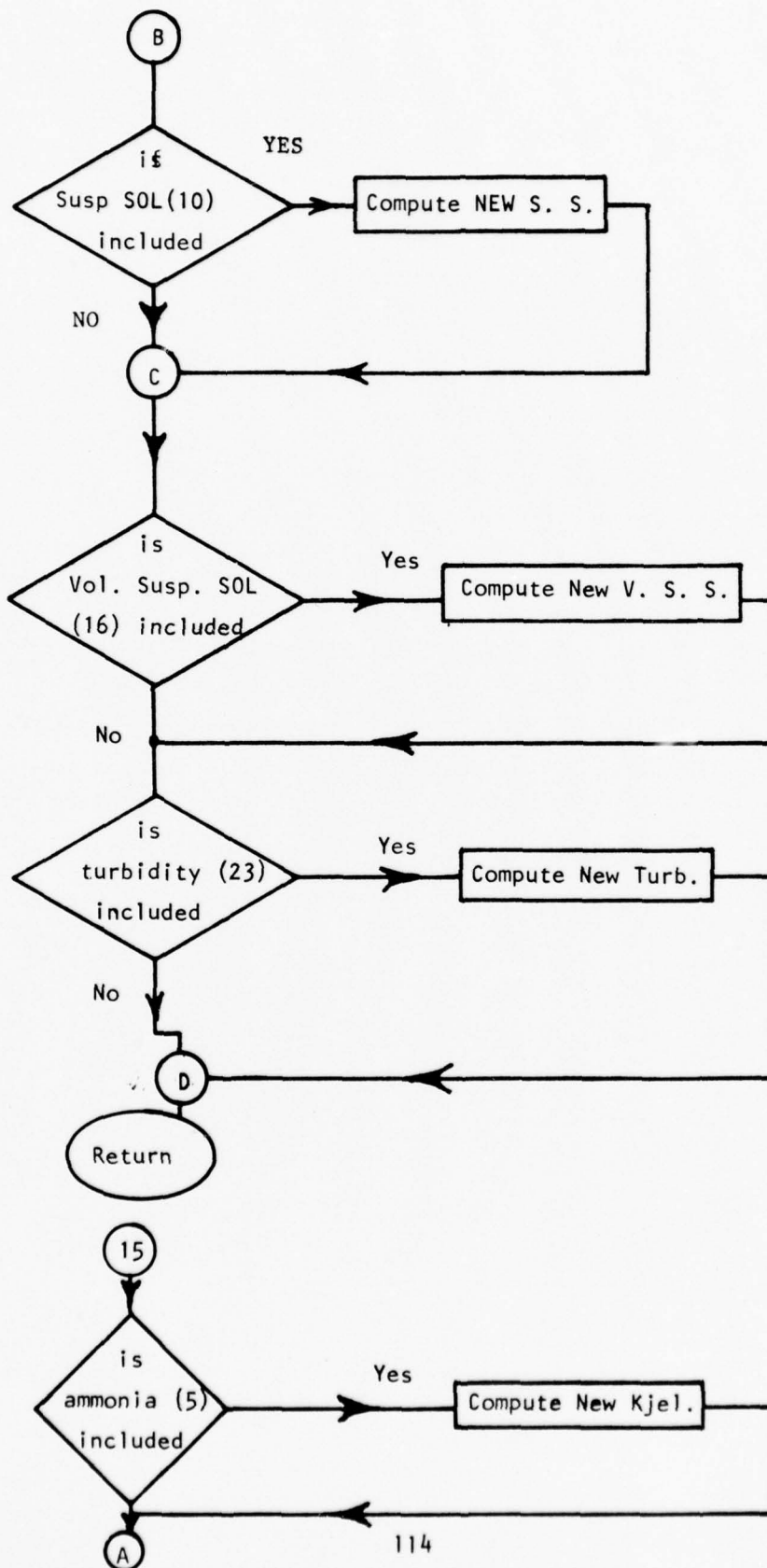


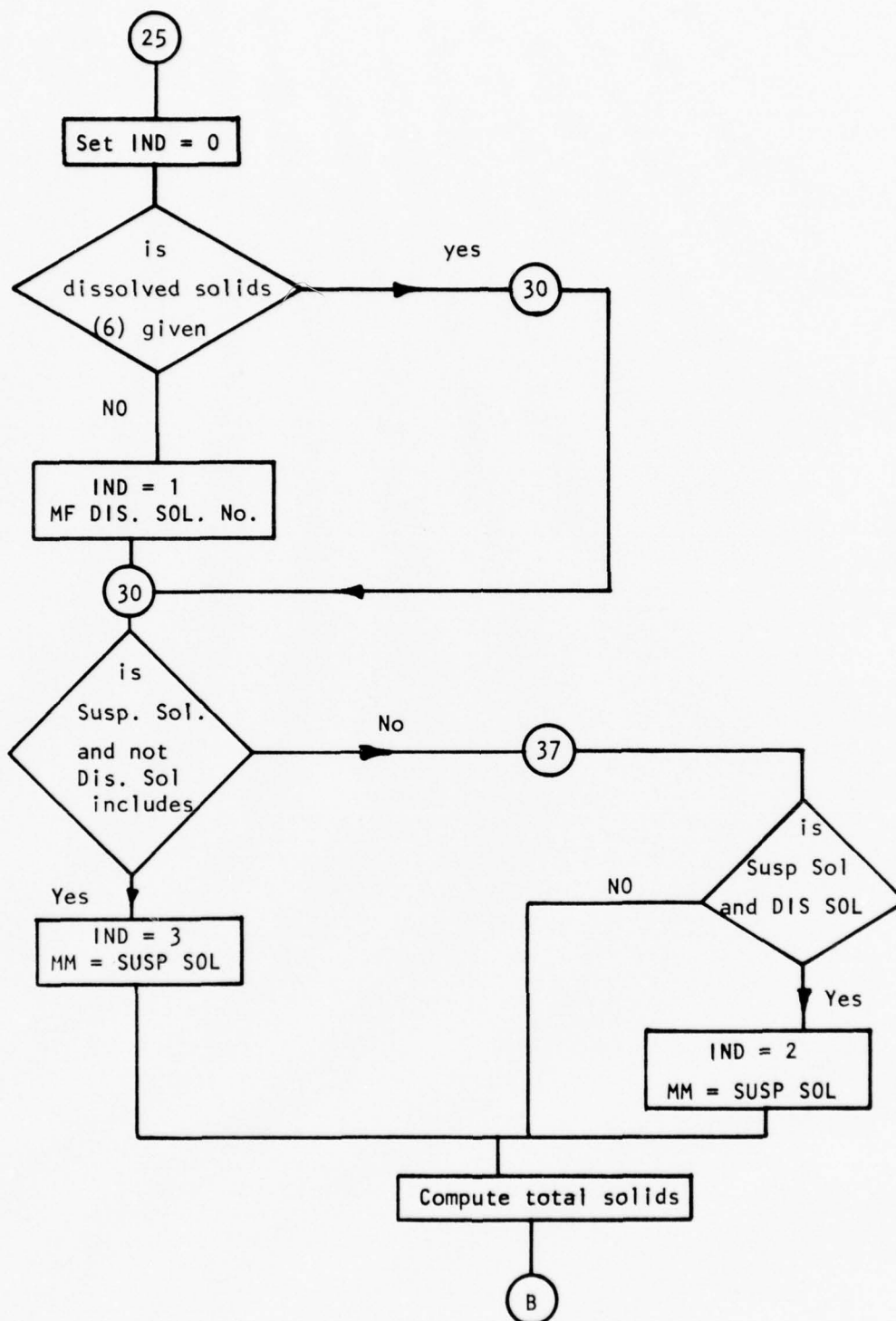




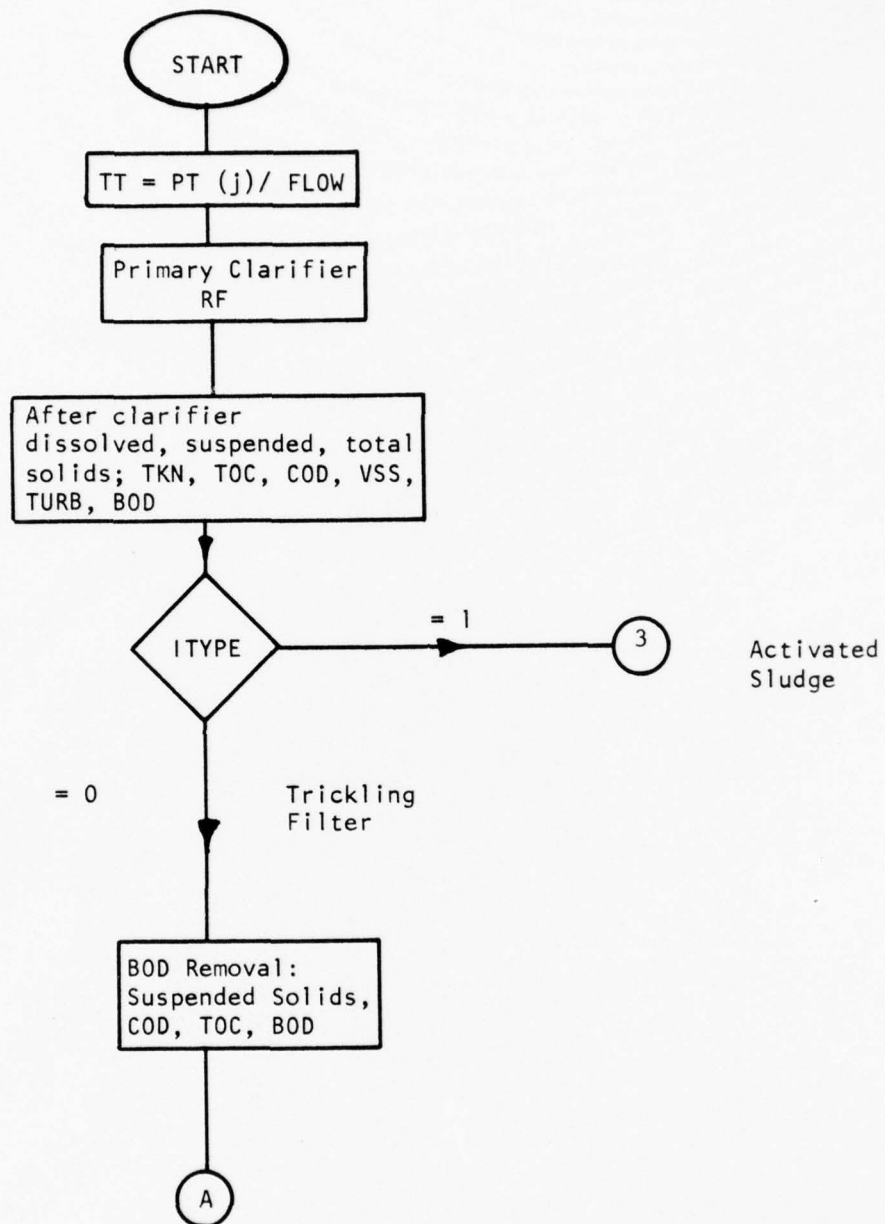
FLOW CHART FOR SUBROUTINE SETTLE

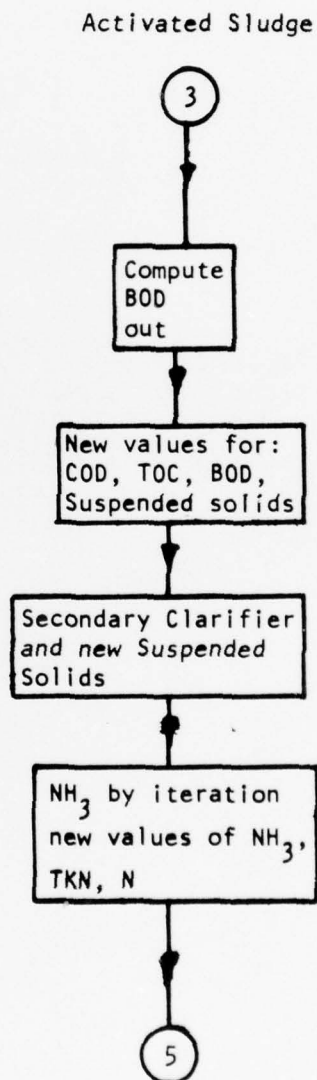
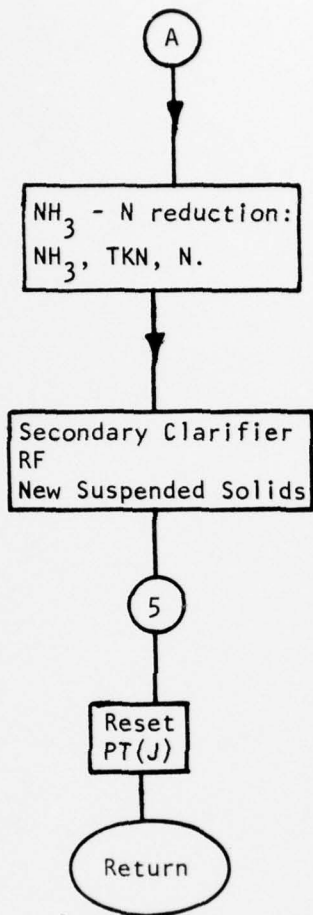


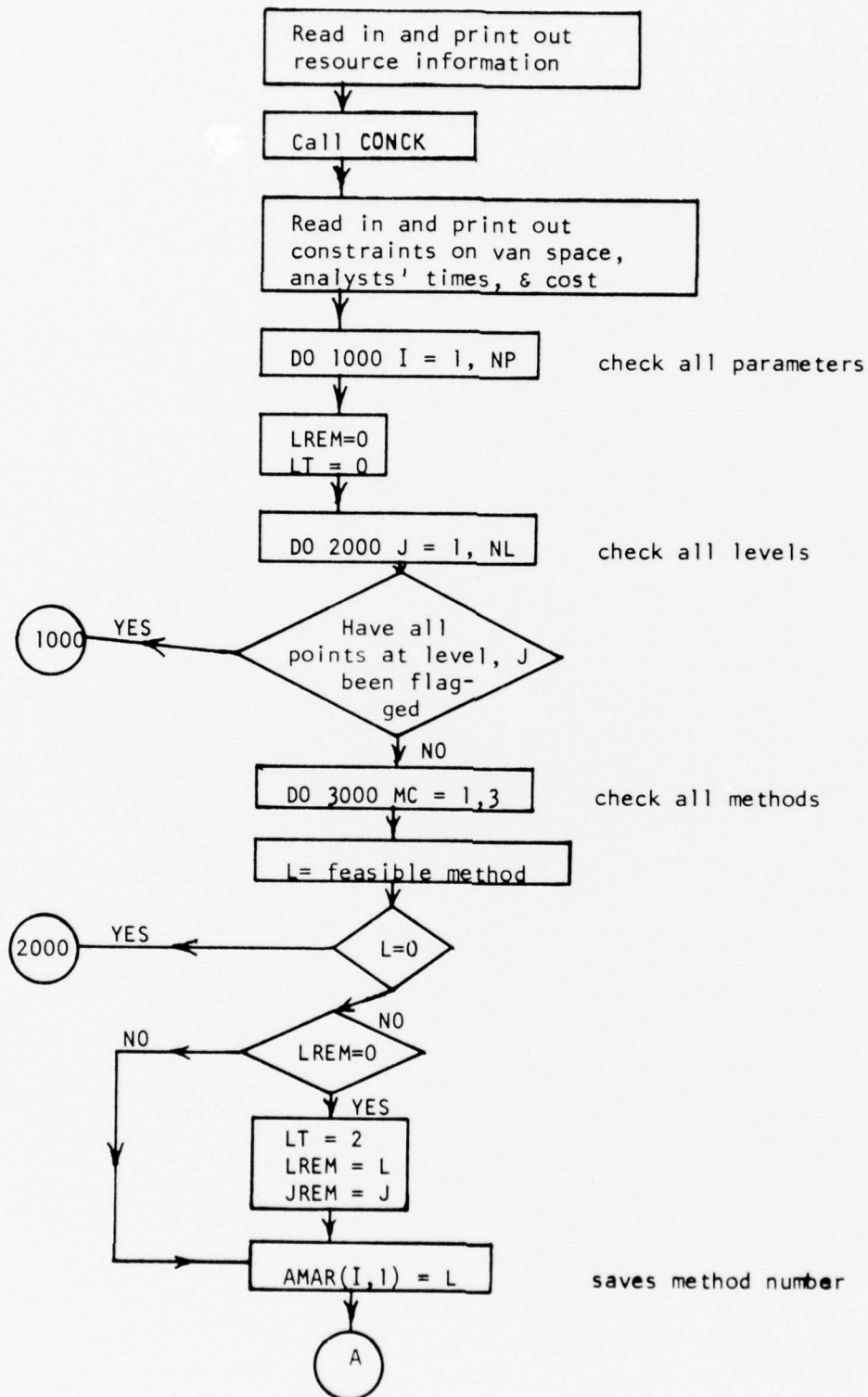


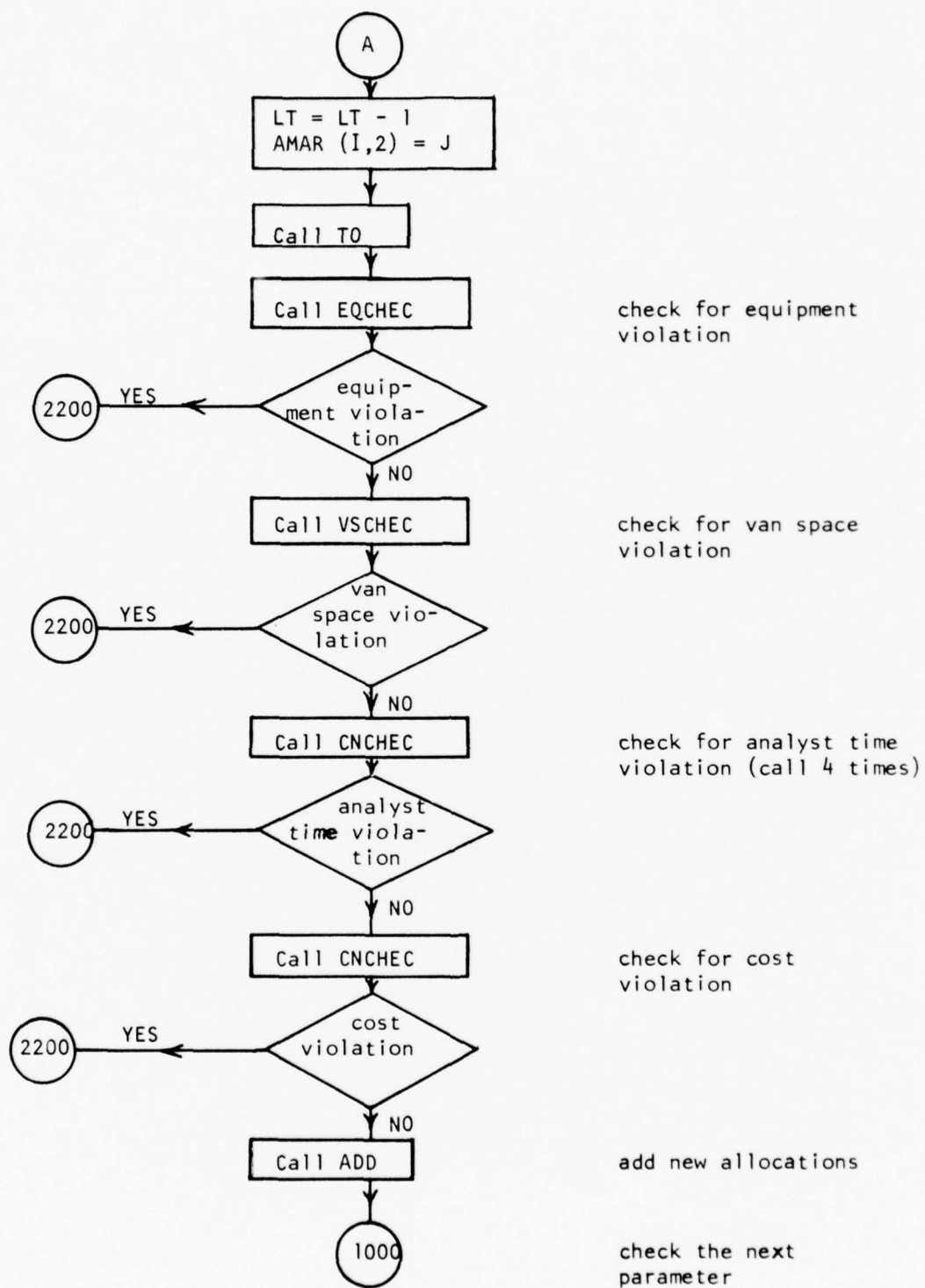


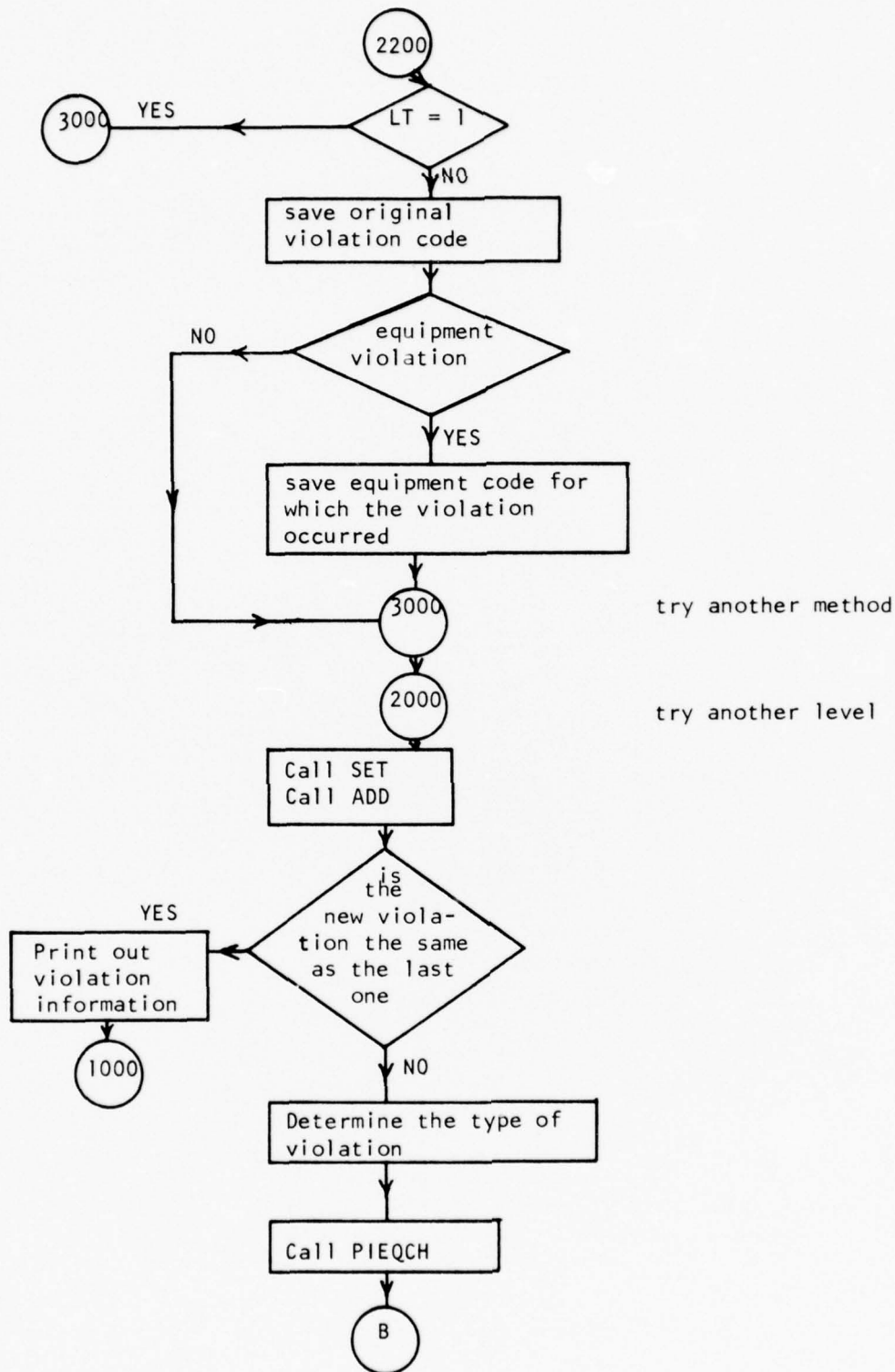
FLOW CHART FOR SUBROUTINE DOME

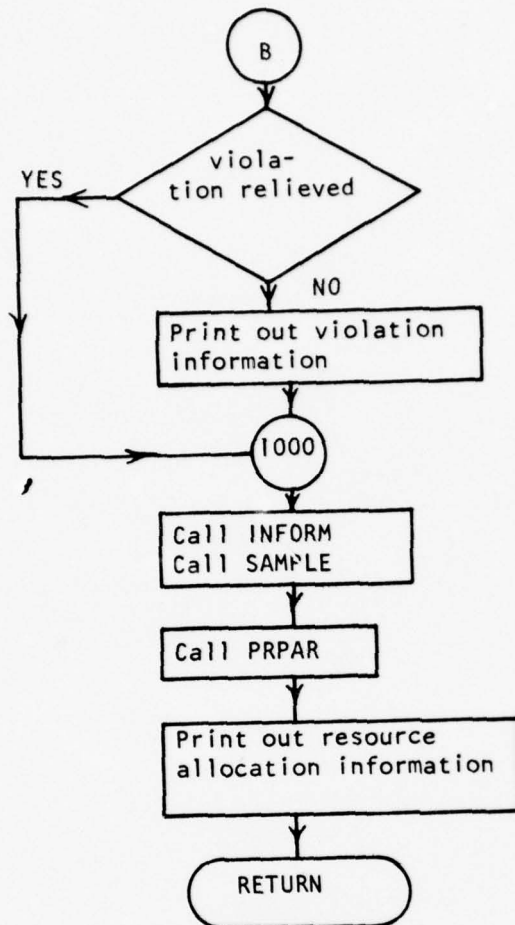






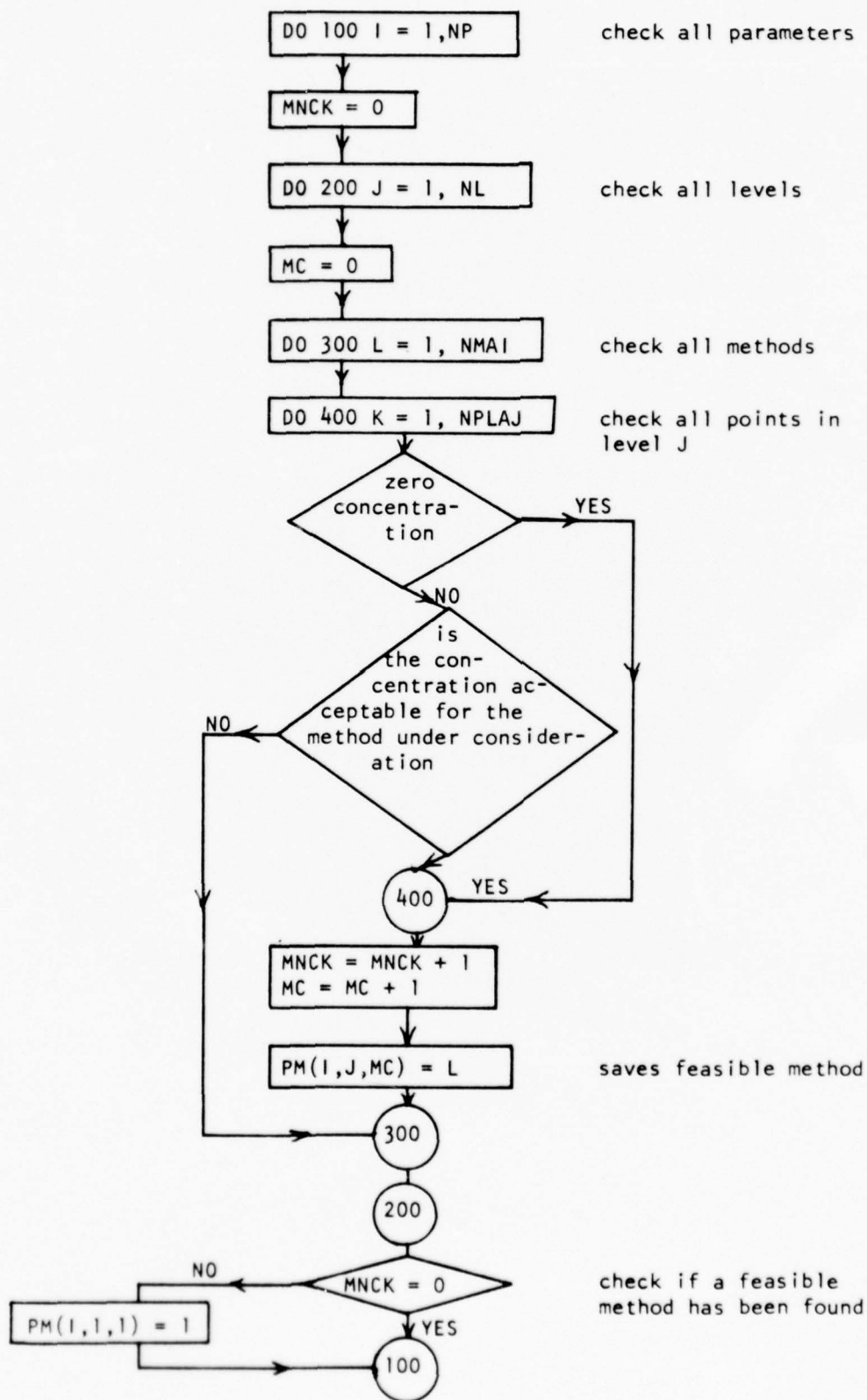




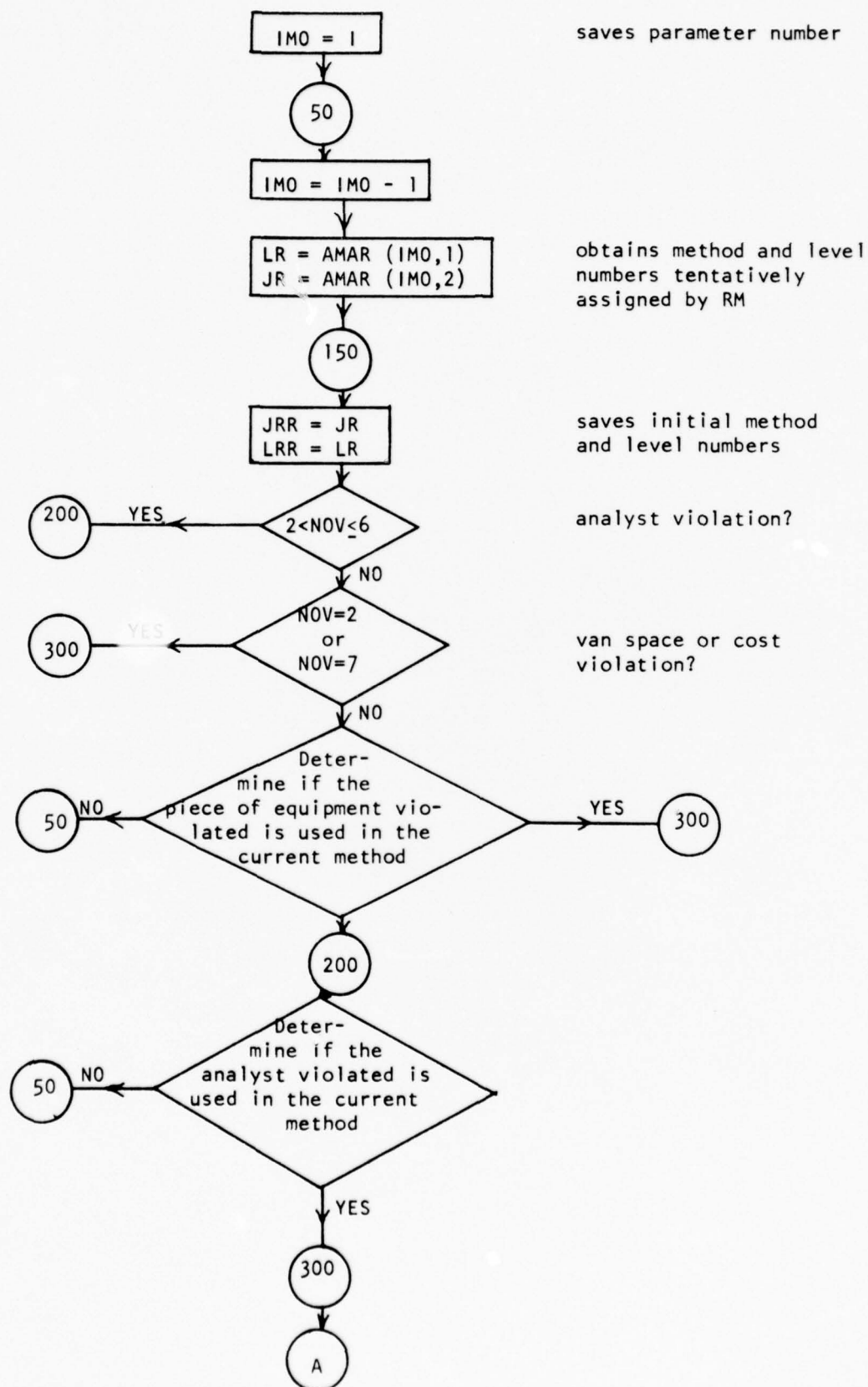


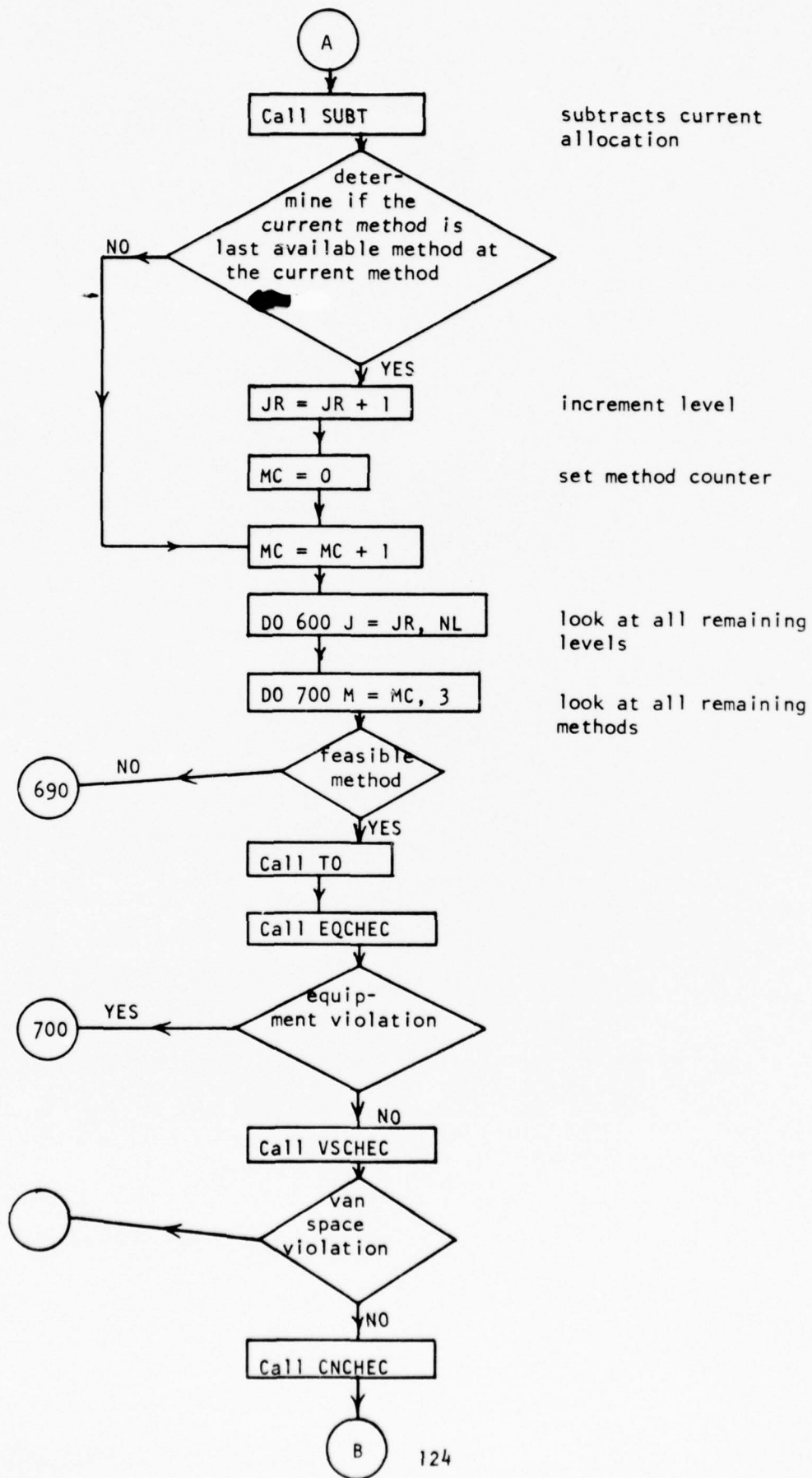
check the next parameter

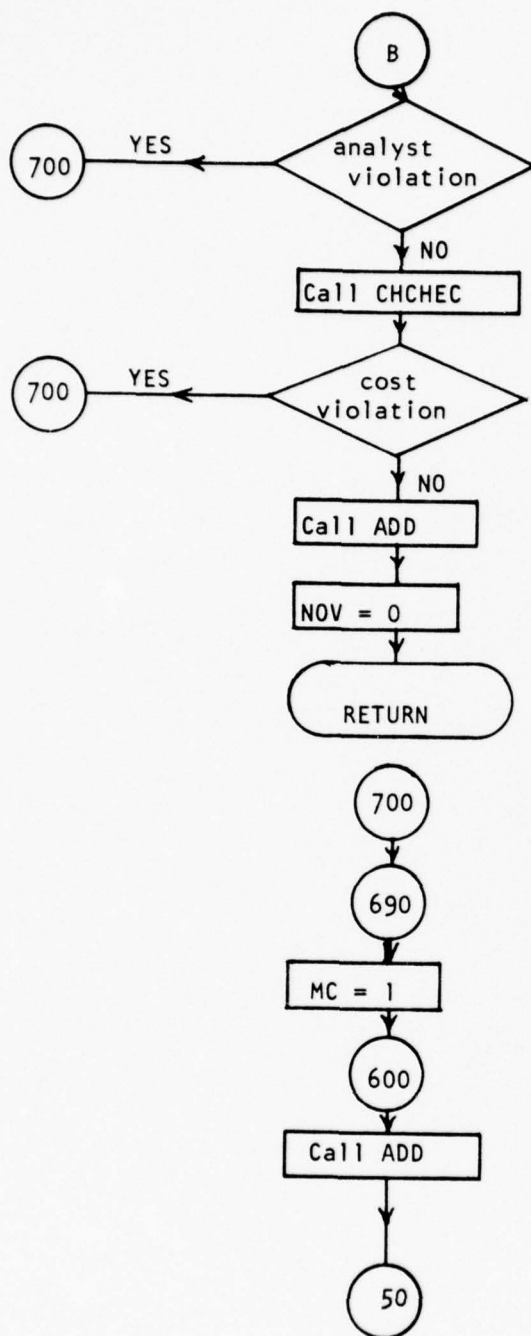
Flow Chart for Subroutine CONCK (feasible method determination only)



Flow Chart for Subroutine PIEQCH







add new allocations

set violation code

continue method search

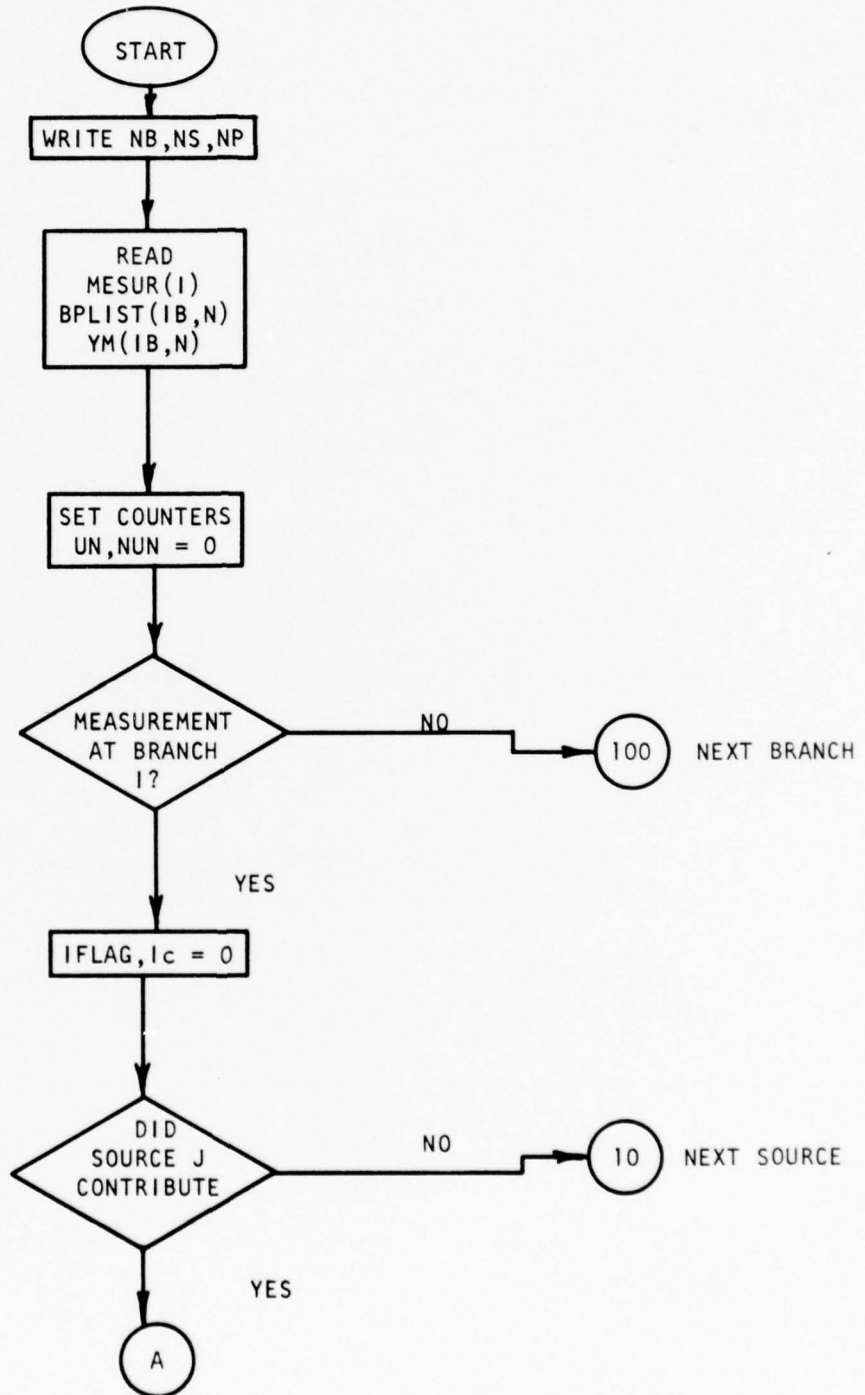
set method counter

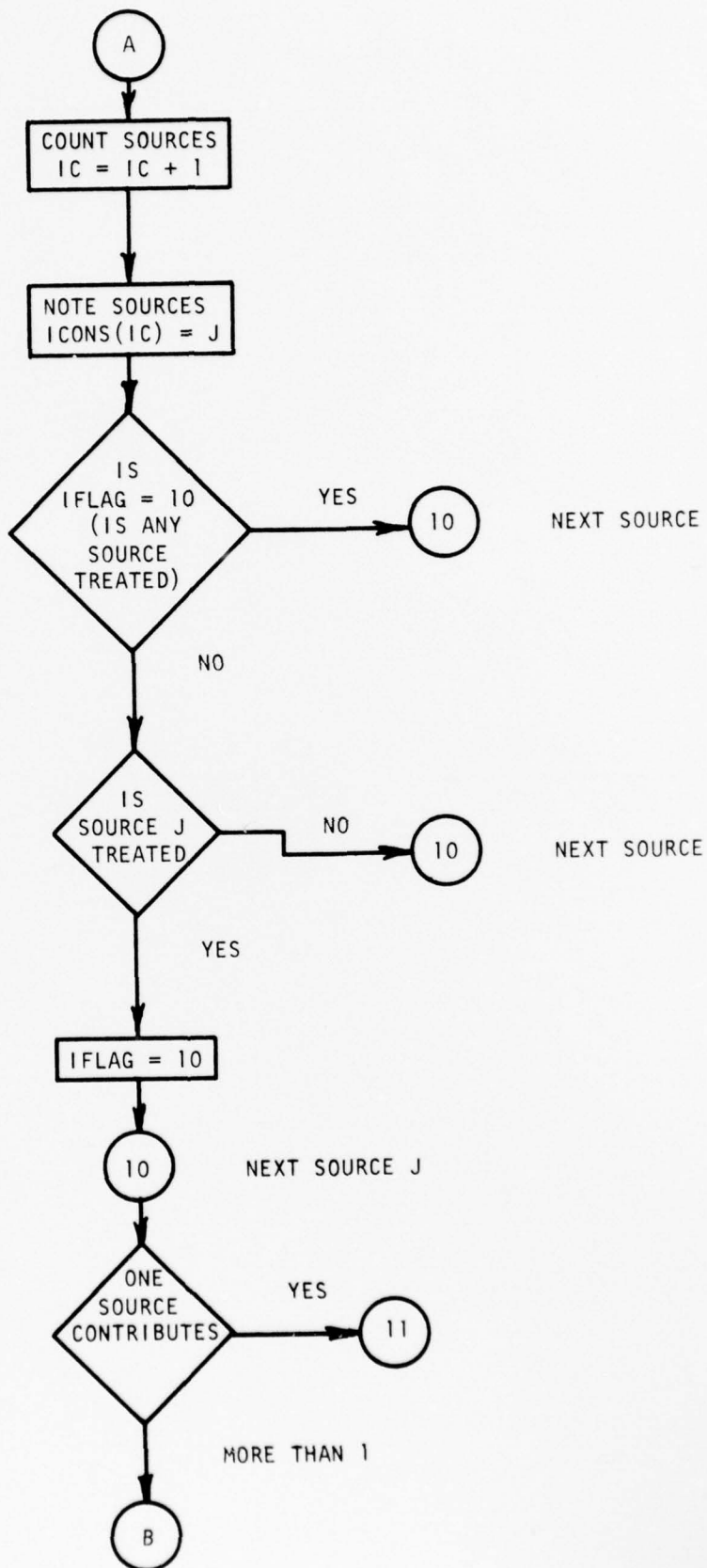
continue level search

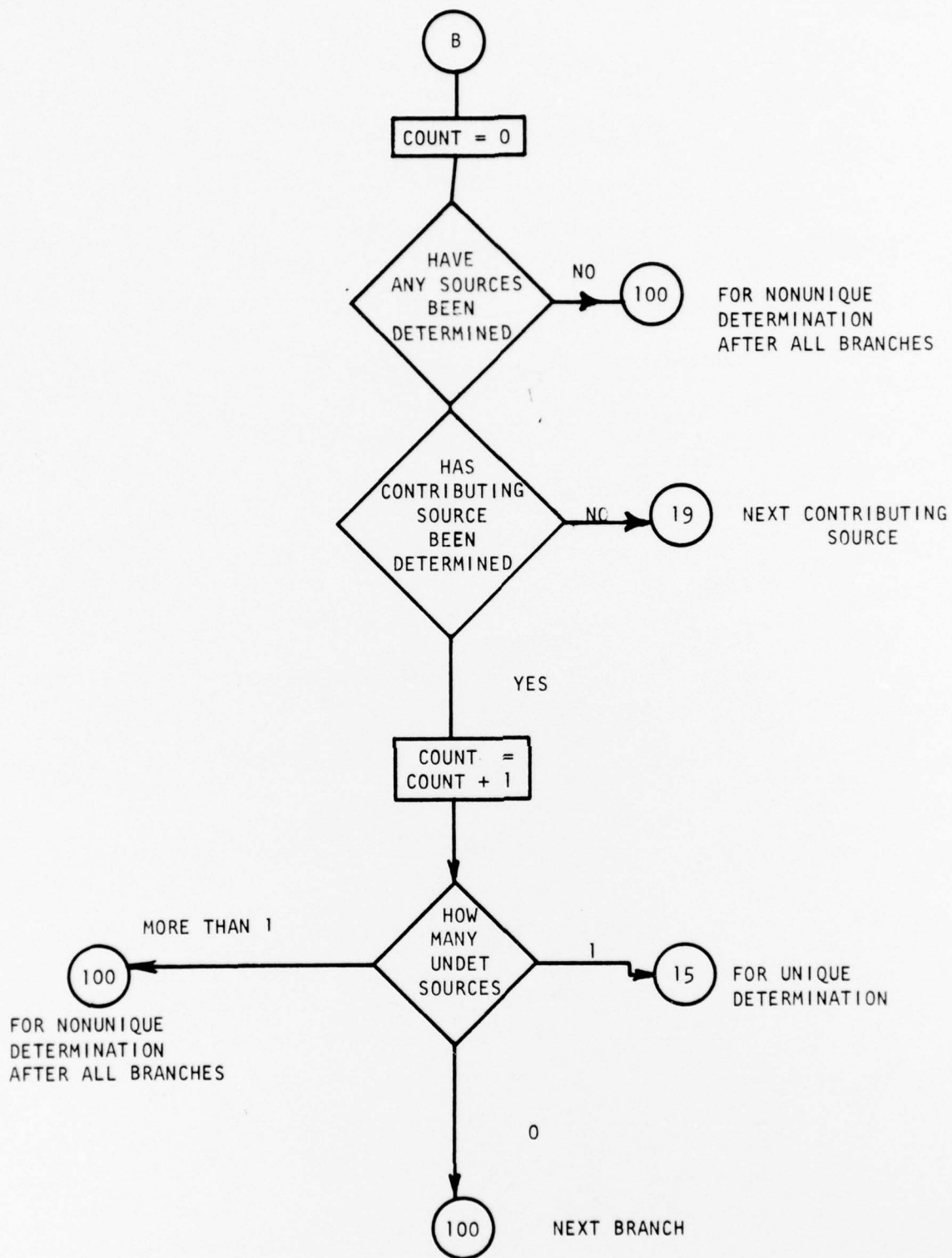
no relief was found for
current parameter so add
previous allocations

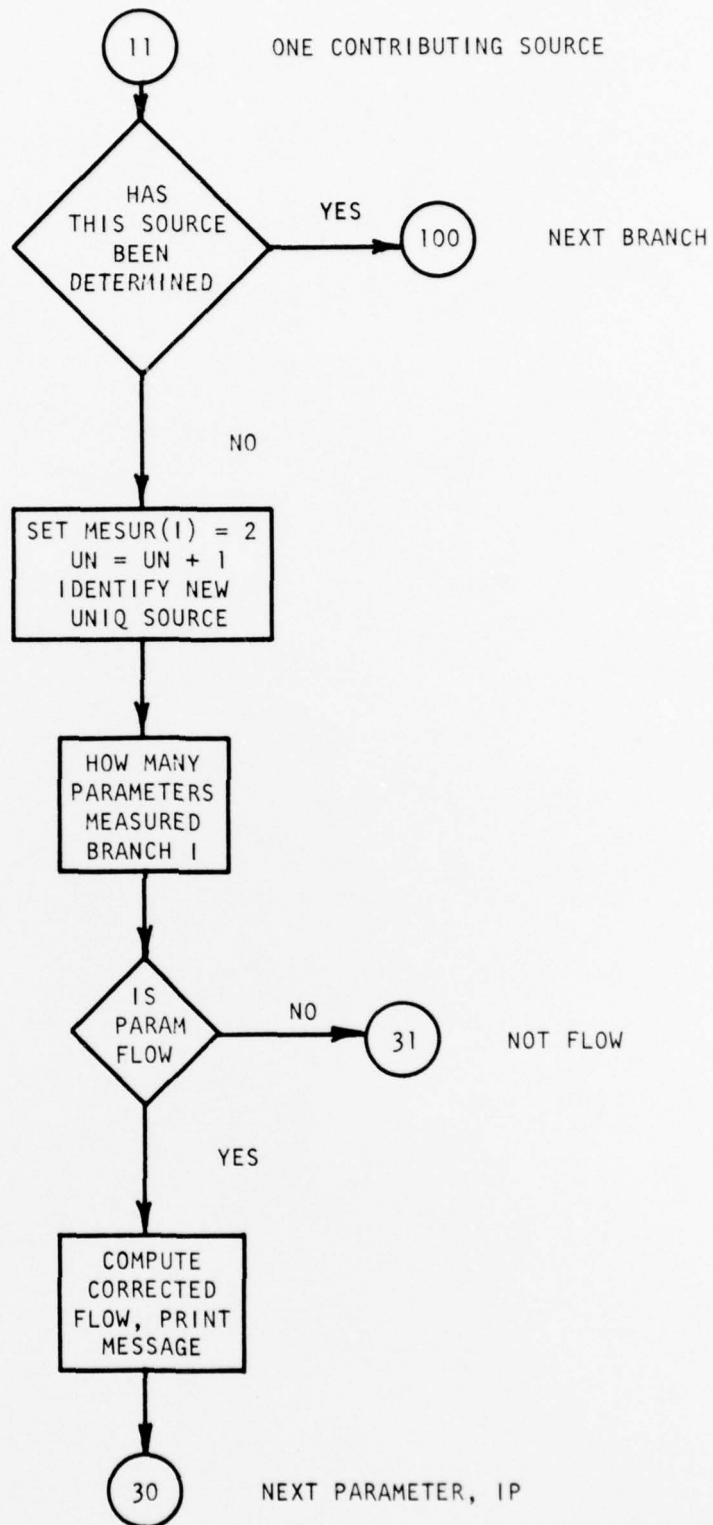
FLOW CHART FOR SUBROUTINE

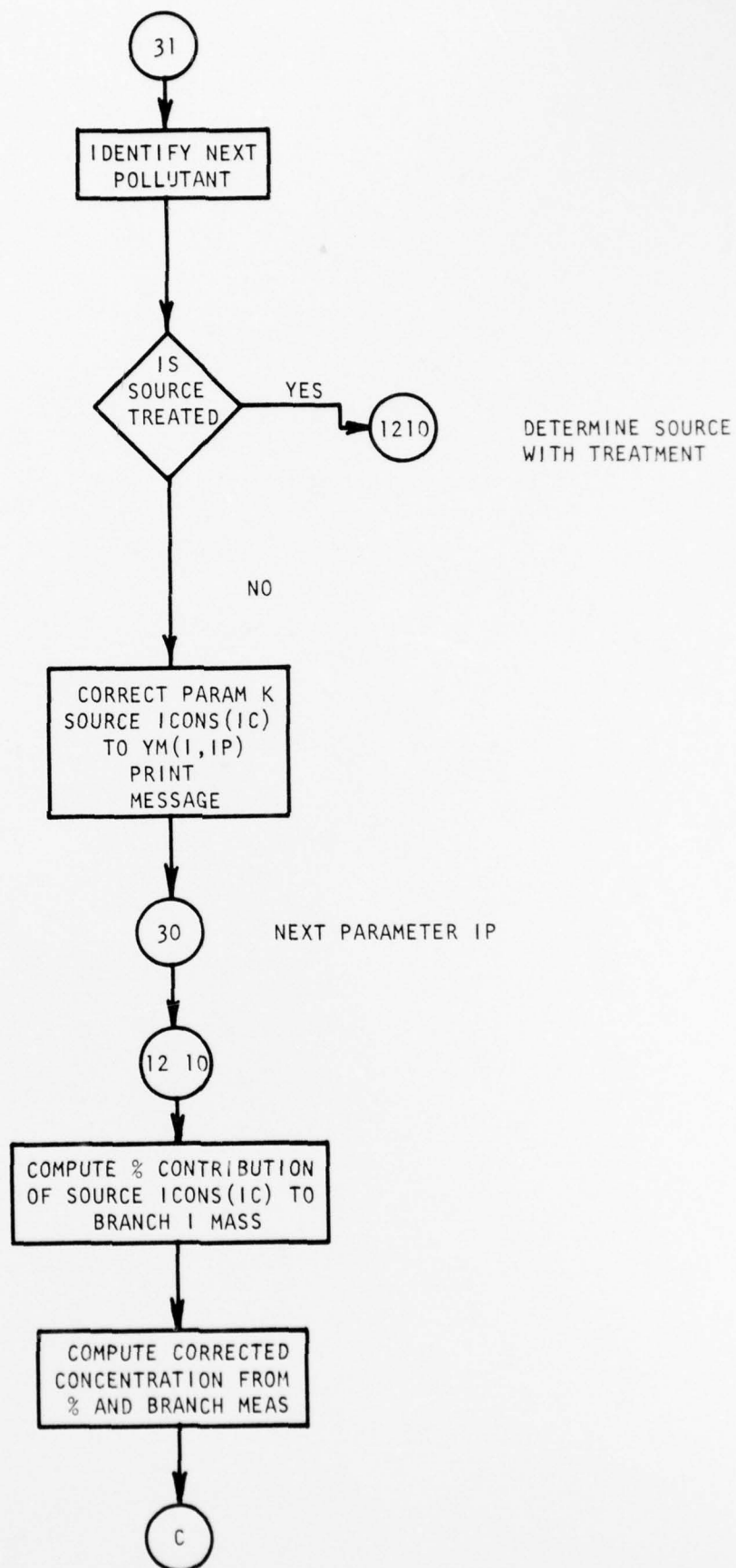
CORRCT

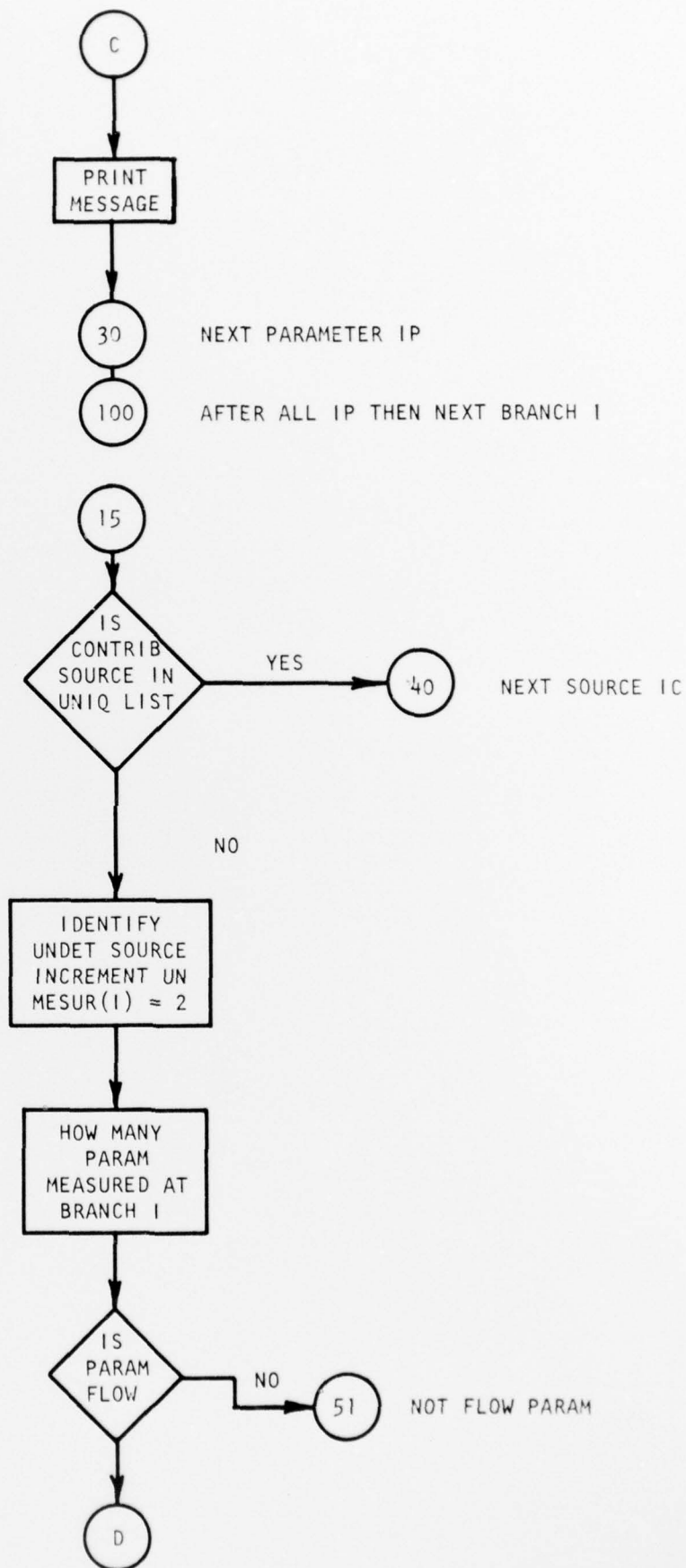


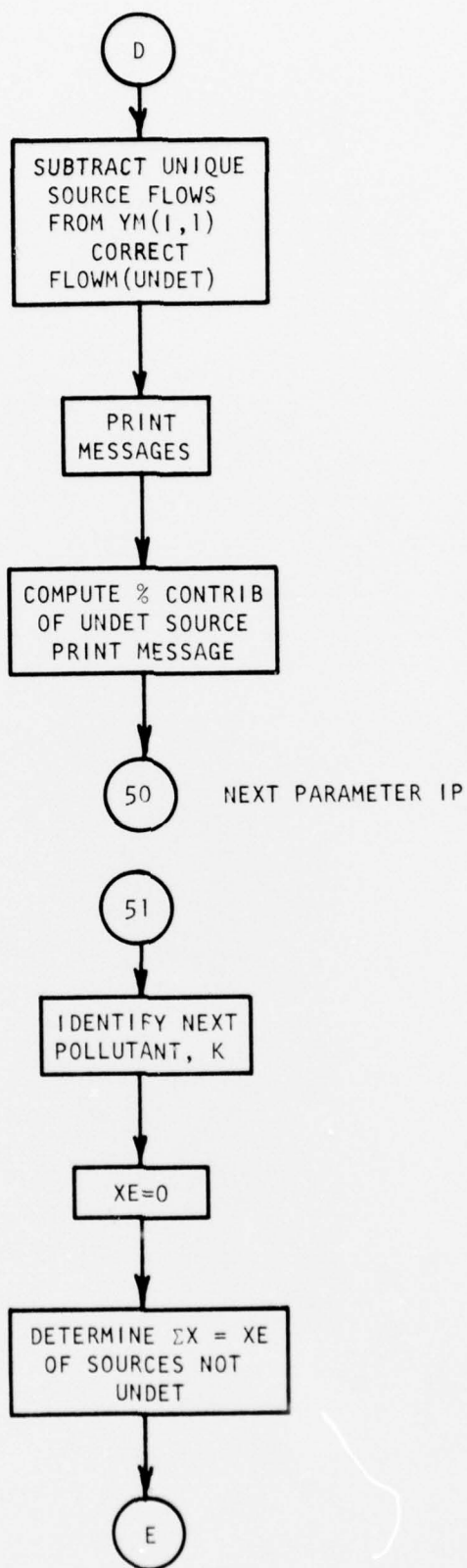


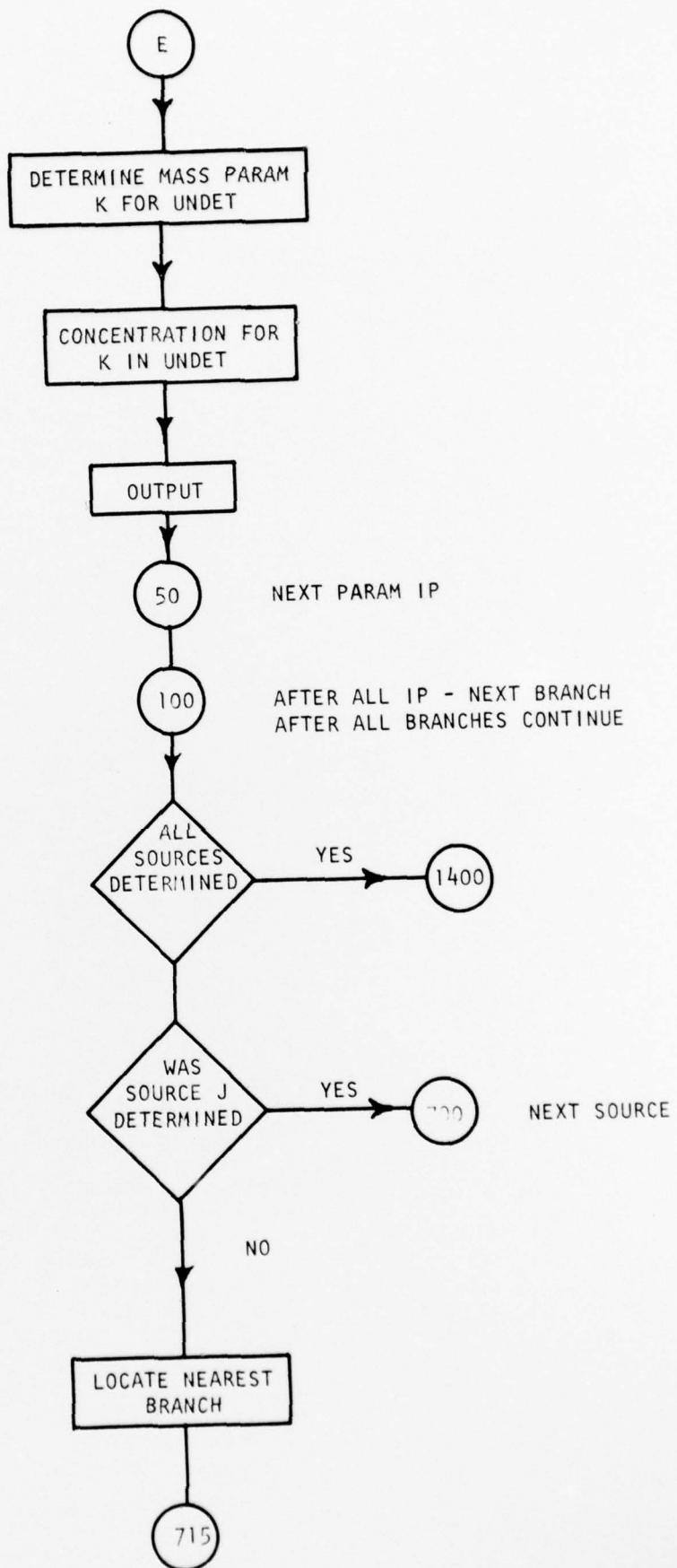


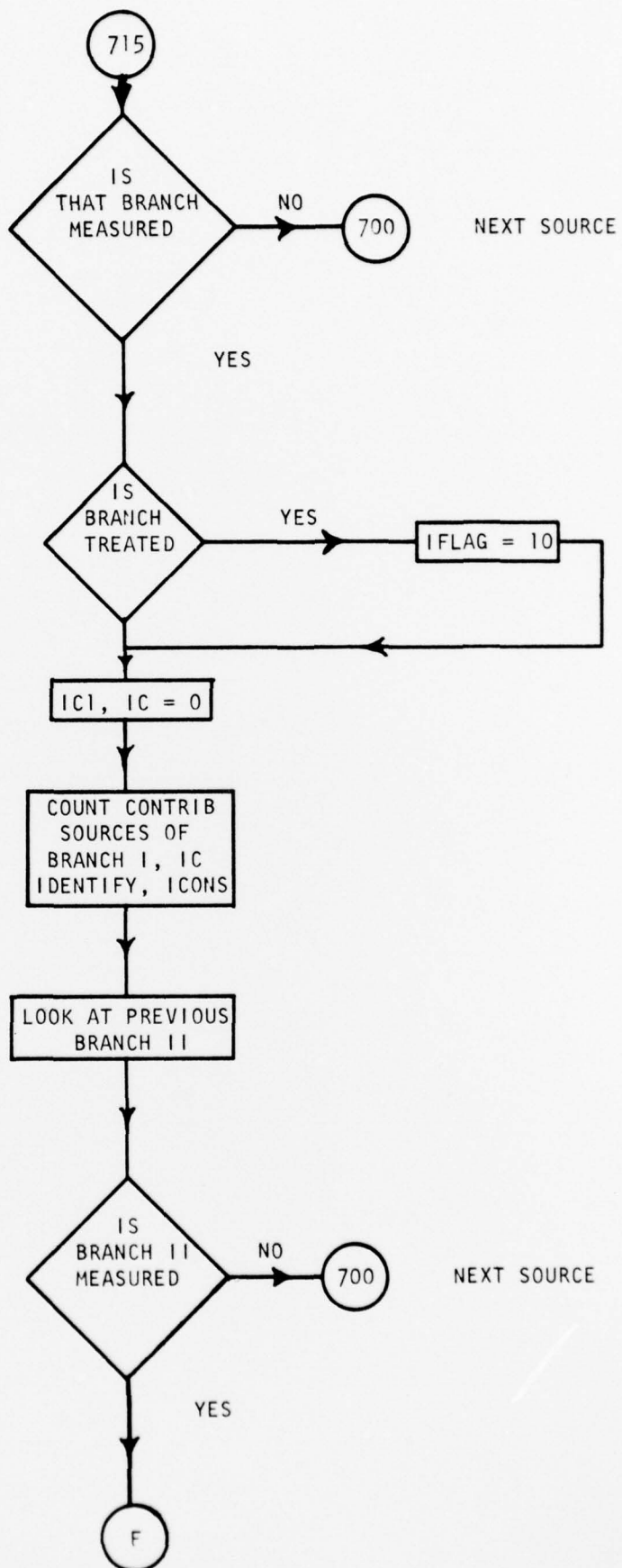


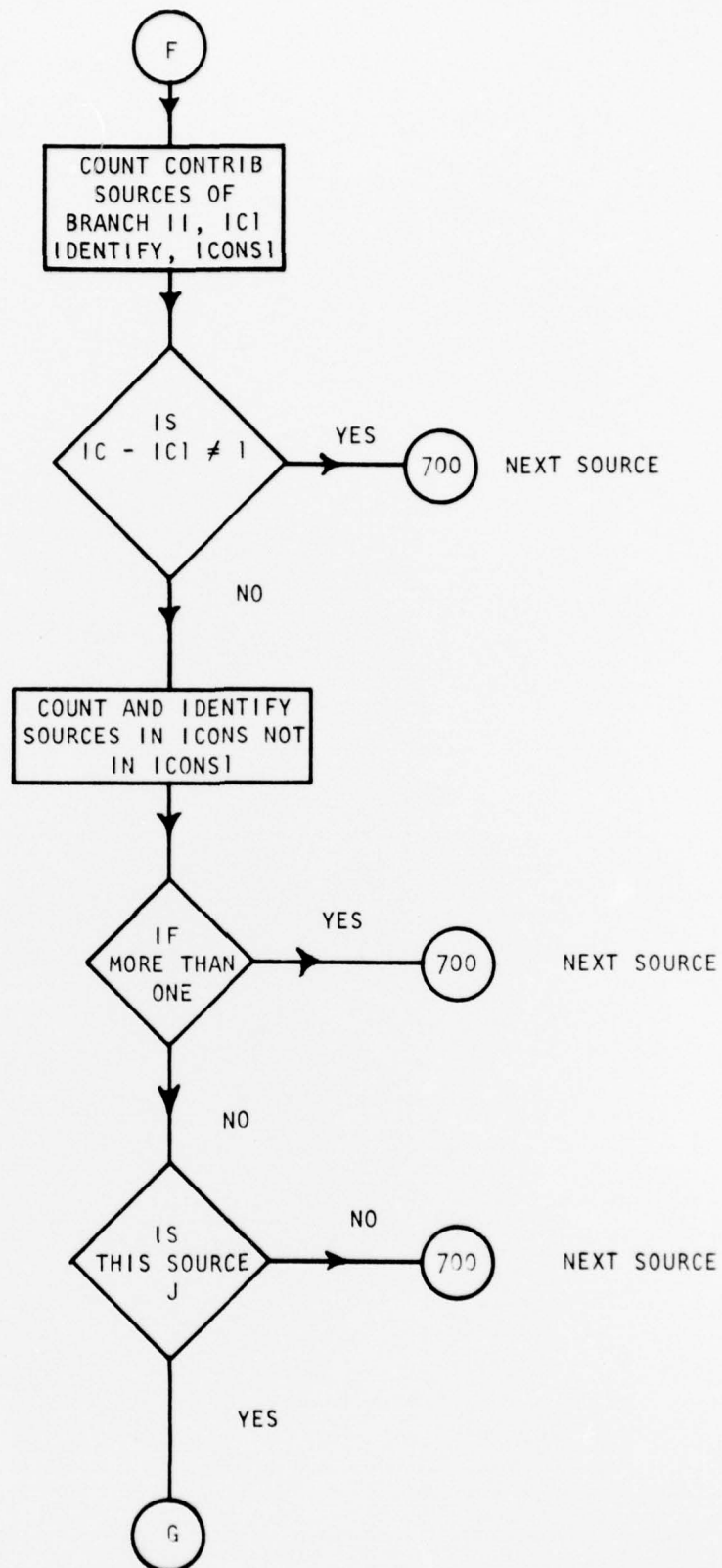


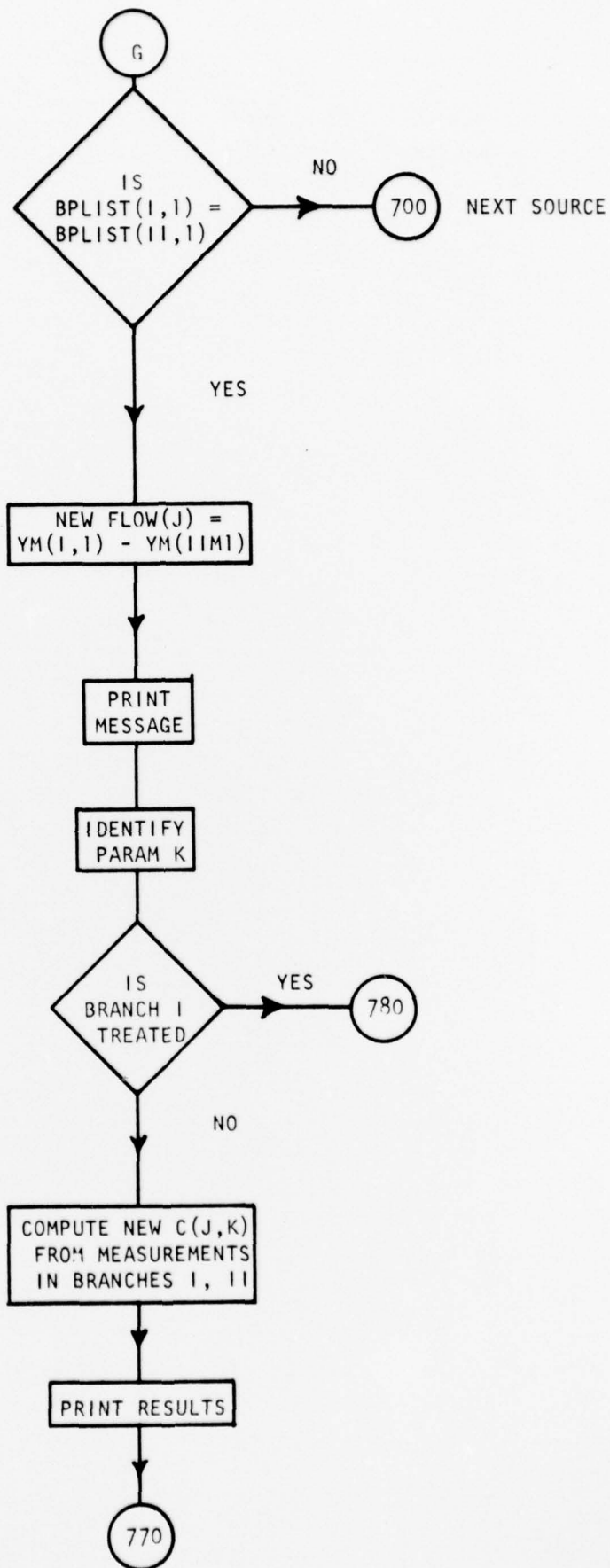


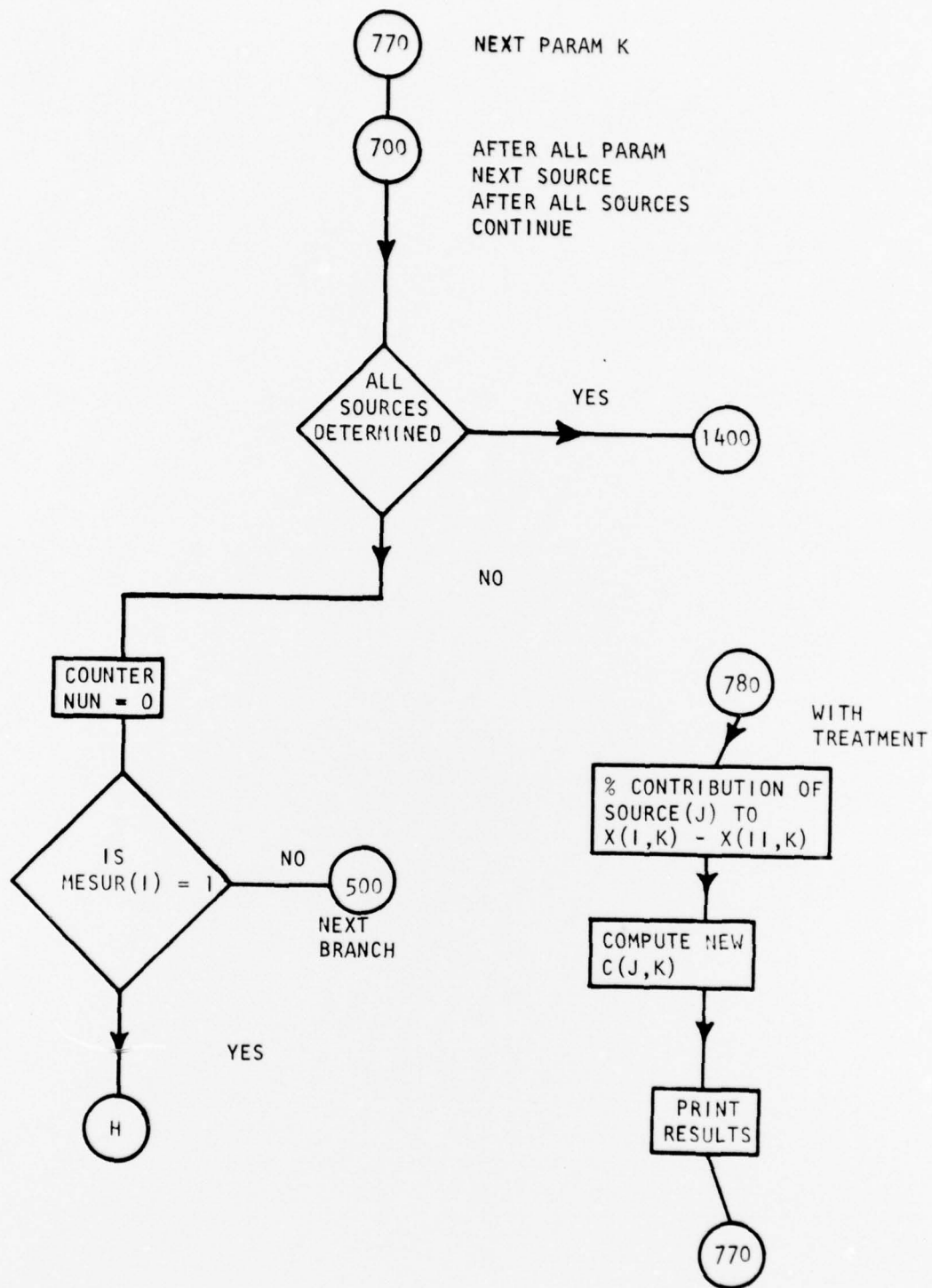


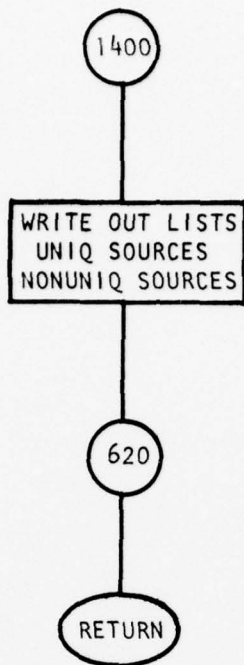


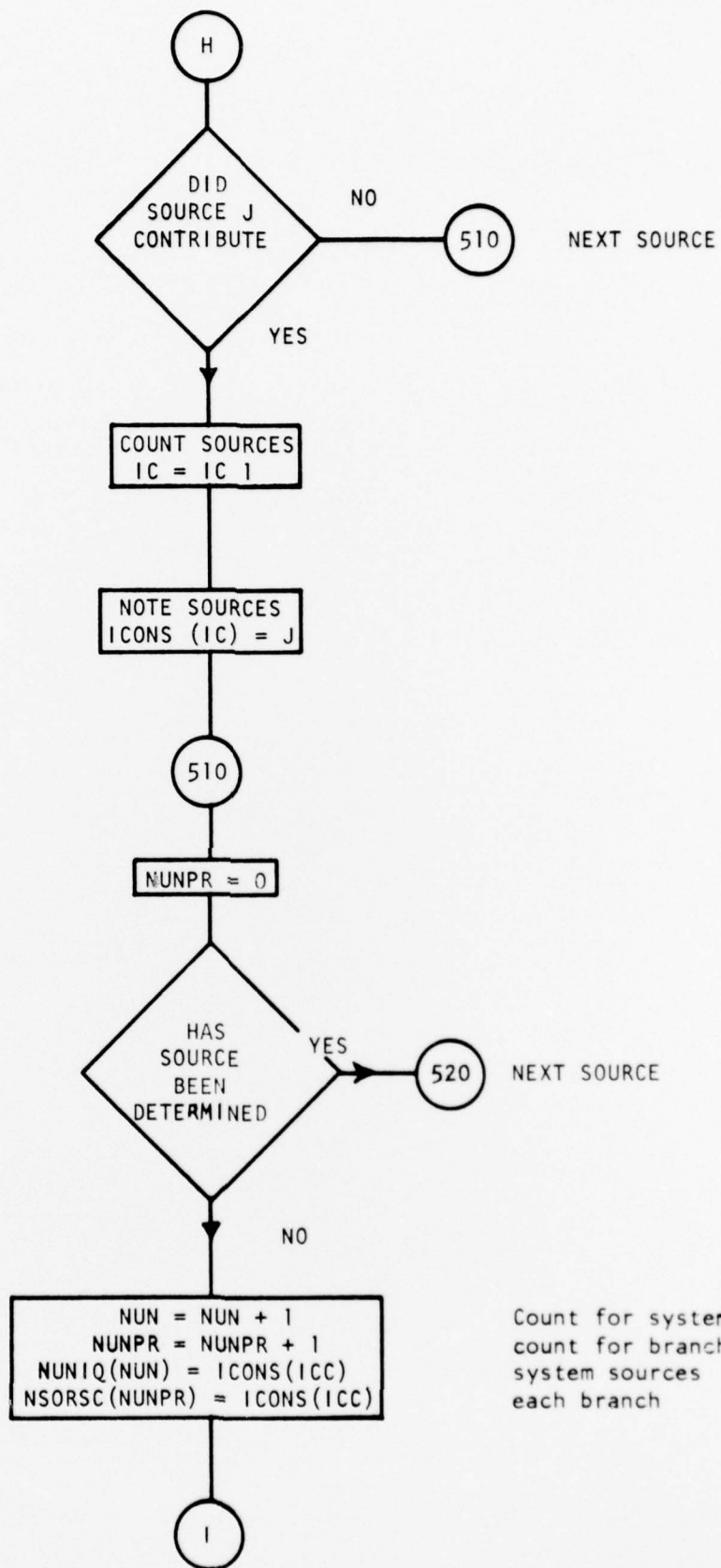


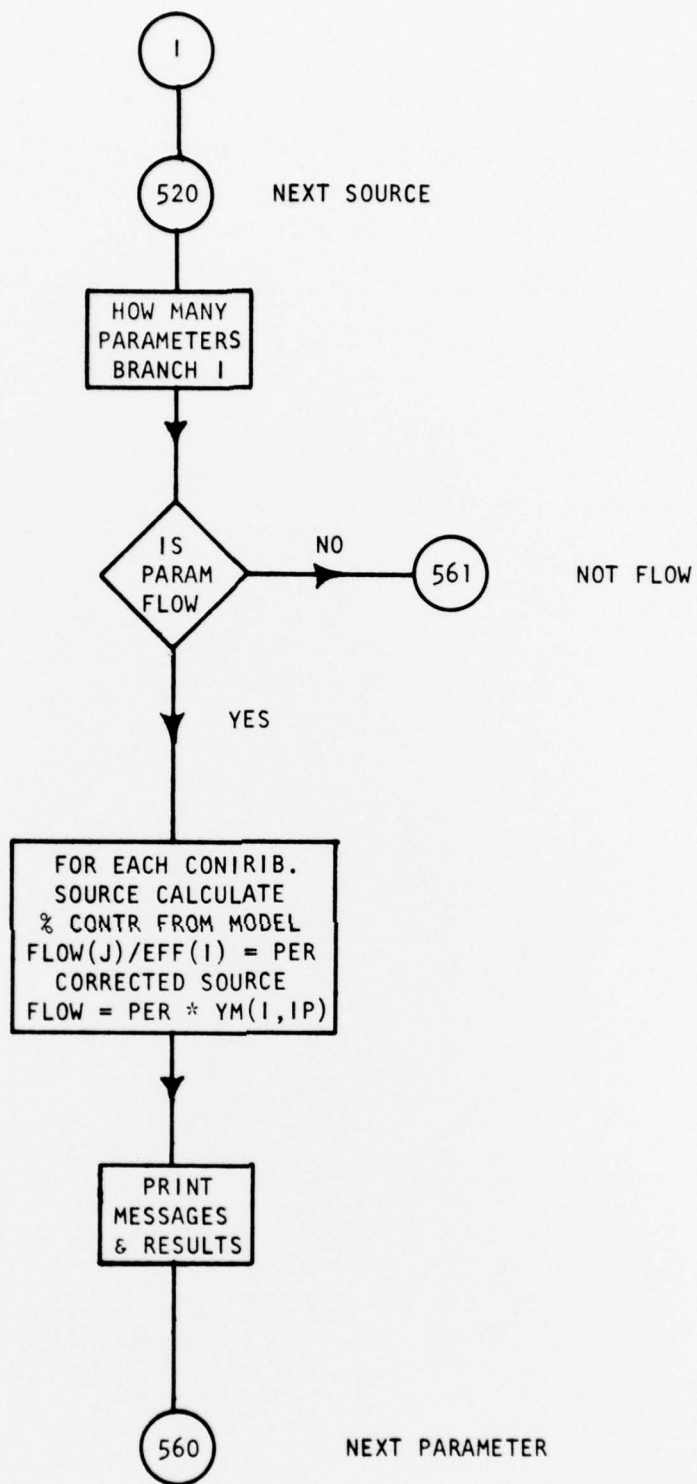












START NEXT PAGE AT 561

